Planning Plausible Human Motions for Navigation and Collision Avoidance

Je-Ren Chen

Department of Computer Science
University College London

January 30, 2014

I, Je-Ren Chen, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

This thesis investigates the plausibility of computer-generated human motions for navigation and collision avoidance. To navigate a human character through obstacles in an virtual environment, the problem is often tackled by finding the shortest possible path to the destination with smoothest motions available. This is because such solution is regarded as cost-effective and free-flowing in that it implicitly minimises the biomechanical efforts and potentially precludes anomalies such as frequent and sudden change of behaviours, and hence more plausible to human eyes. Previous research addresses this problem in two stages: finding the shortest collision-free path (motion planning) and then fitting motions onto this path accordingly (motion synthesis). This conventional approach is not optimal because the decoupling of these two stages introduces two problems. First, it forces the motion-planning stage to deliberately simplify the collision model to avoid obstacles. Secondly, it over-constrains the motion-synthesis stage to approximate motions to a sub-optimal trajectory. This often results in implausible animations that travel along erratic long paths while making frequent and sudden behaviour changes. In this research, I argue that to provide more plausible navigation and collision avoidance animation, close-proximity interaction with obstacles is crucial. To address this, I propose to combine motion planning and motion synthesis to search for shorter and smoother solutions. The intuition is that by incorporating precise collision detection and avoidance with motion capture database queries, we will be able to plan fine-scale interactions between obstacles and moving crowds. The results demonstrate that my approach can discover shorter paths with steadier behaviour transitions in scene navigation and crowd avoidance. In addition, this thesis attempts to propose a set of metrics that can be used to evaluate the plausibility of computer-generated navigation animations.

Acknowledgements

Thanks to my examiners, Jian Jun Zhang and Nadia Berthouze. To my supervisor, Anthony Steed, I am most grateful for his guidance and support. Thanks to my fellow VECG group members at UCL and my friends in the VFX community in London, and particularly to James Tompkin, Martin Parsley, KK Yeung, and Richie Xu, for their generous help.

Finally, thanks to my family in Taiwan, especially my mum, without whom I would never be able to start my doctoral research let alone finish this thesis.

Contents

1 Introduction **12**

1.1 Motivation 13

1.2 Contribution 15

1.3 Overview 17

2 Related Work **18**

2.1 Humanoid-character Motion Planning 18

2.2 Graph-structured Motion Database 21

2.3 Query-based Character Navigation 24

2.4 Distributed and Data-driven Crowds 26

2.5 Synthetic Human Motion Evaluation 30

2.6 Summary 31

3 Single-character Navigation **33**

3.1 Path Finding and Following 33

3.2 Environment-aware Motion Graphs 36

3.2.1 Full-body Collision Detection 37

3.2.2 Joint Cost Function 38

3.3 Implementation 39

3.3.1 Randomised Motion Graph Search 39

3.3.2 Potential-field Path Finding 40

3.4 Experimental Results 41

3.4.1 Two-corner Turning 43

3.4.2 Object-cluttered Space 44

3.5 Summary 45

4 Multi-character Navigation **54**

4.1 Interdependency of Interpersonal Interactions 55

4.2 Path Interference Perturbations 56

4.3 Implementation 57

4.3.1 Path Initialisation 58

4.3.2 Path Interference . . . 58
4.3.3 Path Perturbation . . . 60
4.3.4 Refinements . . . 62
4.4 Non-degeneracy and Completeness . . . 62
4.5 Results . . . 63
4.5.1 Four-person Crossing . . . 64
4.5.2 Multiple Crossing . . . 66
4.5.3 Central Locking . . . 68
4.5.4 Station Concourse . . . 69
4.6 Discussion . . . 70

5 Plausibility Metrics **72**
5.1 Anomalous Collision Avoidance Behaviours . . . 73
5.2 Implementation . . . 74
5.2.1 Resistance . . . 74
5.2.2 Jerkiness . . . 76
5.2.3 Proximity . . . 78
5.2.4 Synchronicity . . . 80
5.3 Further Analysis . . . 84
5.3.1 Scene Navigation Strategies . . . 84
5.3.2 Levels of Collision Detection . . . 85
5.3.3 Levels of Congestion . . . 86
5.4 Discussion . . . 86

6 Conclusion **89**
6.1 Summary . . . 89
6.2 Future Work . . . 90

A Supplementary Videos **93**
A.1 Environment-aware Motion Graphs . . . 93
A.2 Path Interference Perturbation . . . 93

Bibliography **94**

List of Figures

1.1 Two common human motion generation problems addressed in this thesis: scene navigation (left) and interpersonal collision avoidance (right). 12

1.2 Two consequences of decoupled motion search: (left) a trajectory generated using conservative collision assumption, where red areas are obstacles and black areas are the offset to keep the trajectory away from obstacles; and (right) captured motions fail to fit on the trajectory, which result in an animation that penetrates into the obstacle. 14

1.3 Two common anomalous behaviours caused by local navigation models: (left) dispersed crowd formations and inter-penetration between individuals due to the failure of balancing contradictory steering forces; and (right) frequent course and pace changes among the crowd due to the congestions caused by local collision-avoidance strategies. Please refer to the video in Appendix A.2. 15

2.1 Free-space representations: (a) visibility graphs build a network using vertexes of the obstacle. (b) Navigation meshes decompose the free space into convex polygons. (c) Roadmaps are constructed to connect points distributed in the free space. (d) Potential fields are computed to draw a subject into the goal. 19

2.2 A joint position-based distance matrix computed between two motions. Dark regions indicate poses that are numerically similar. 22

3.1 Path finding and following. 35

3.2 The schematic view of the environment-aware motion graph. 36

3.3 The *dynamic bounding circle* associated with each frame. Notice how the radius changes to tightly fit each pose. 37

3.4 Each pose is labelled with a corresponding supporting-foot phase such as left-foot (green), right-foot (indigo), and double-support (orange). 39

3.5 Potential-field path finding: (a) Initial location (green) and goal location (orange) among obstacles (red). (b) The gradient circle of the potential field centred at the goal. (c) Potentials are stored in discretised free-space cells. (d) Potentials are used as heuristics to search for a chain of cells (in orange) connecting to the initial location. (e) A trajectory is converted and straightened from the chain. 40

3.6 An example of planning a trajectory using potential fields. The black cells are determined using a obstacle-growing technique (Lozano-Pérez and Wesley, 1979) to keep the trajectory away from obstacles. 41
3.7 A snapshot sequence of two-corner turning. The distance between PFF (indigo) and EMG (orange) gradually increases after each corner. 44
3.8 The potential-field path-finding in two-corner turning. The obstacles (red) are expanded (black regions) to keep the path away. 45
3.9 The results of the two-corner turning test scene: (a) the distance between PFF and EMG (indicated by the green dashed lines) gradually increased. (b) EMG requires less travel distance and travel time and lower transition rate and transition cost than PFF. 47
3.10 A snapshot sequence of a grid of boxes. PFF (indigo) traverses a zig-zag path (see Figure 3.11 while EMG stays close to the shorter diagonal line.) 48
3.11 The potential-field path finding in a grid of boxes. The expansion of the obstacles forces the path finder to choose an erratic zig-zag path. 48
3.12 The results of the grid-of-box test scene in object-cluttered space. 49
3.13 A snapshot sequence of an example object-cluttered space test scene (Figure 3.21). Whilst both PFF and EMG can travel around obstacles in this experiment, EMG is able to find a shorter path through smaller clearances between obstacles. 50
3.14 The potential-field path finding in an example randomised object-cluttered space test scene (see Figure 3.21 for the comparisons of the results). 50
3.15 The results of random config. 1. 51
3.16 The results of random config. 2. 51
3.17 The results of random config. 3. 51
3.18 The results of random config. 4. 52
3.19 The results of random config. 5. 52
3.20 The results of random config. 6 52
3.21 The results of random config. 7 53
3.22 The results of random config. 8 53
3.23 The results of random config. 9 53

4.1 A cropped region of a collision incident resolution during mutual avoidance optimisation: (a) Initial paths. (b) Local incident detection in path interference. (c) Rollback mechanism. (d) Path re-planning in path perturbation. The result is either (e) collision-free, or (f) collided with another re-planning at $T + t', t' \geq 1$. 59
4.2 Iterations of path re-planning during four-person crossing. 61
4.3 A comparison between simulated trajectories (dots) and fitted motions (solid lines) in the four-person crossing test scene. My PIP method does not require to approximate a trajectory in order to generate full-body animation. 64

4.4 The results of the four-person crossing crowds. (a) While PIP chooses slightly longer routes than RVO2 and Pettré et al., the crowd of PIP arrives at the goals almost as soon as Pettré et al.'s. (b) The speed profile of PIP is smooth and uses a similar spectrum of speed as Pettré et al.'s pedestrian model. 65

4.5 Snapshots of the resulting animations for the multiple crossing crowds. 66

4.6 The results of the multiple crossing crowds. (a) PIP plans shorter routes and animations that arrive at the goals sooner. (b) RVO2 shows sudden changes of speed indicating anomalous motions. 66

4.7 Snapshots of the resulting animations for the central locking crowds. 67

4.8 The results of the central locking crowds. (a) OpenSteer shows larger goal deviation (hatched areas), indicating the failure to arrive close to the goals due to the repulsive collision avoidance strategy. (b) Both RVO and OpenSteer show drastic speed changes while the crowd of PIP can avoid each other with a smoother and faster speed profile. . . 67

4.9 The convergence of the costs over the incremental optimisation steps of the central locking simulation with our algorithm. Notice that the algorithm trade-off travel time and travel distances for minimising goal deviations and collision errors over iterations. 68

4.10 Snapshots of the resulting animations for the train station scene. 69

5.1 Benchmark results for energy consumption. The resistance ratio of the energy used to avoid collisions are compared to indicate possible excess energy in the navigation animations. 76

5.2 Normal (a) and tangential (b) jerk, the rate of the change of acceleration, measured in the "four-person crossing" experiment. The grey dashed lines are benchmark upper and lower limits measured from the "walking motion" database. Notice that while RVO2 (green) and OpenSteer (blue) cause high jerks beyond the upper and lower limits, PIP (red) remains within or close to the limits. 77

5.3 Benchmark results for change of behaviours. The grey dashed lines are the jerkiness (the standard deviation of jerk) measured in the "walking" and "running" motion database. RVO2 show very high jerkiness in both directional and speed changes in more crowded situations, indicating hyperactive and impulsive collision avoidance behaviours. The results of OpenSteer show that while it can maintain a steadier speed, it requires frequent change of direction in order to avoid collisions. 78

5.4 Benchmark results for interpersonal clearances. The grey dashed lines are maximum and minimum bounding radius measured in the "walking" motion database. OpenSteer shows severe inter-penetrations that violate the minimum bounding radius. RVO2 remains close to the maximum bounding radius, which could show excessive interpersonal spaces between individuals. The crowd of PIP can avoid each other closely without violating the minimum bounding radius. 79

5.5 Pair synchronicities measured in four-person crossing: (top) The velocity profiles, (middle) the curvilinear acceleration profiles, and (bottom) the heat maps of pair synchronicity of the individuals, with dark red being highly synchronised movement. RVO2 shows identical synchronisation while OpenSteer shows little sign of coordination between pairs. PIP shows intermittent coordinations when pairs are avoiding each other. 80

5.6 Pair synchronicities measured in multiple crossing. Pairs are arranged in the chronological order as each individual appears in the scene. RVO shows blocks of dark red regions indicating several long periods of identical synchronisation. The pairs in OpenSteer are highly synchronised in speed (t-sync) but not in direction (n-sync), resulting a disordered pair synchronicity with little coordination. The crowd of PIP shows patterns of occasional coordinations. 81

5.7 Pair synchronicities measured in central locking. The results suggest that overall the crowds are less synchronised/coordinated than crowds in multiple crossing as there are more individuals converge in the middle at the same time. RVO still produces periods of identical synchronised movements between many of its pairs. 82

5.8 Histograms of pair-wise synchronicity. The distribution of pair synchronicity in PIP are situated between the range of $0.5 - 0.8$, showing consistent levels of coordination. The data distribution varies noticeably with different crowd settings in OpenSteer. RVO2 produces more hyper-synchronised pairs ($\bar{s} > 0.95$) than PIP and OpenSteer. 83

5.9 Benchmark results for group synchrony. Each bar indicates the overall synchronicity with the solid region highlighting the portion of the hyper-synchronised pairs. The synchronicity of PIP gradually decreases as the crowd becomes denser in the centre of the scene. RVO2 and OpenSteer both create hyper-synchronised groups. 84

5.10 Benchmark comparison of "object-cluttered space" experiments in Chapter 3. The dashed lines are the benchmark jerkiness measured in the "walking" motion database. The resistance measurement shows that EMG requires less effort to avoid obstacles. While the measurements of EMG are mostly lower than PFF, both methods can produce low-jerkiness animations within the benchmarks. 85

5.11 Benchmark comparison between three levels of collision detection: (level 0) maximum bounding circles, (level 1) dynamic bounding circles, and (level 2) full-body collision detection. Reducing the precision of collision detection results in higher resistance and jerkiness, and the decrease of the coordination between pairs. 86

5.12 Benchmark comparison between different levels of congestion. The metrics correctly indicate that the effort to avoid collision increases as the scene becomes more crowded, so as the n-jerkiness due to more frequent changes of direction. Pair coordination also becomes rarer as shown in the synchronicity measure. 87

List of Tables

3.1 The motion capture data used to construct the “walking” motion graph. 43

4.1 Collision errors before and after root-fixing. 65
4.2 The motion capture data used to construct the “running” motion graph. 69

Chapter 1

Introduction

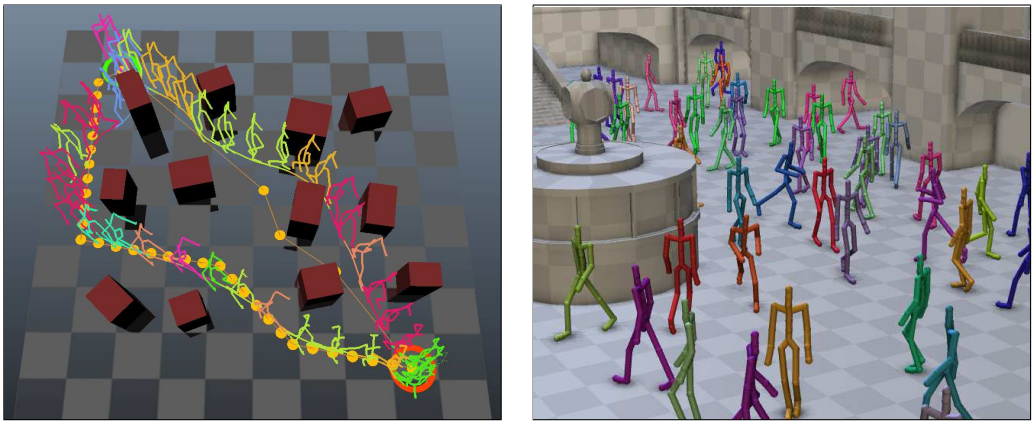


Figure 1.1: Two common human motion generation problems addressed in this thesis: scene navigation (left) and interpersonal collision avoidance (right).

From best-selling feature films and video games to multi-million civil engineering constructions and urban planning projects, computer-generated human motions have become increasingly important for a wide range of applications. Both the research communities and the industry are in need of new techniques for the simulation, visualisation, and analysis of human behaviours. Over the past decade, researchers and practitioners have been able to record the performance of multiple real human subjects in great detail using the latest motion capture technologies. They can also analyse and manipulate captured data to generate smooth and natural-looking human motions with state-of-the-art motion database techniques. Despite the advances in the acquisition and reproduction of realistic human movements, there are, however, situations where the interactions between the computer-generated human motions and their surroundings do not match our own experience in analogous real-world scenarios. Therefore, the results are implausible from the human-behaviour perspective. Unlike physics-based animation, human motions are not only restrained by their complex kinematics and physical properties but more importantly driven by intentions and by interactions with the surroundings. Thus, it remains a very challenging problem to generate plausible human motions.

Being able to plan motions to avoid collisions is crucial in the creation of plausible navigation animations. Currently, collision avoidance is considered at a very coarse level which is based on simplified collision models and approximated obstacle-avoidance strategies. This often results in anomalies such as erratic travel paths and impulsive movements caused by excessive course and pace changes. Thus, the resulting navigation animations do not always look plausible as they fail to portray a human-like behaviour when avoiding obstacles. This becomes a greater issue when we are required to create high-fidelity character animations and to simulate anomaly-free human behaviours. The aim of this thesis is therefore to facilitate collision avoidance towards higher accuracy in the generation of navigation animations with the aim of reducing anomalous collision avoidance behaviours.

In this thesis, I investigate common anomalous collision avoidance behaviours in computer-generated human motions and present new methods for creating plausible character navigation animations. These methods include a query-based motion planning technique that generates full-body navigation animation in complex environments (Chen and Steed, 2011) and an incremental optimisation framework for planning navigation and collision avoidance for multi-character animations (Chen and Steed, 2014). I also presented a set of plausibility metrics that quantitatively evaluate navigation animations in order to detect anomalous collision avoidance behaviours.

1.1 Motivation

The conventional approach for generating human-character navigation animations considers goal navigation and collision avoidance as two separate stages. The first stage, *motion planning*, is to determine the free space of a path between the initial and the goal locations; and then the second stage, *motion synthesis*, is to create an animation that travels through the free space along this path. Usually there exists more than one path leading to the goal as any animation that starts at the initial location, moves within the free space, and arrives at the goal location is a valid solution. One common way to choose among these solutions is to find the one that minimises certain costs such as the travel time and travel distance to the goal. This is because it portrays the intention to accomplish a certain task more clearly. Also, it implies this solution requires less effort and hence is more plausible from the biomechanical point of view.

To realise this two-stage motion generation, past work often formulates the first stage as finding the shortest path using a simplified shape instead of the entire body of the virtual human, with the assumption that an animation can be found to traverse this path in the second stage (Kuffner, 1998; Bandi and Thalmann, 1998; Pettré et al., 2003). This is because planning full-body motion is a difficult problem as the kinematics and dynamics of human body are complicated and hard to simulate. Since low-dimensional robotics motion planning is a well-studied problem, one common practise is to simplify the virtual human as a 2-D omnidirectional mobile entity, such as a moving circle, and then to apply path finding algorithms to obtain the shortest path, a 2-D trajectory on the ground, to the goal. Then, in the second stage, motion capture data is exploited to approximate this trajectory with appropriate motion clips to

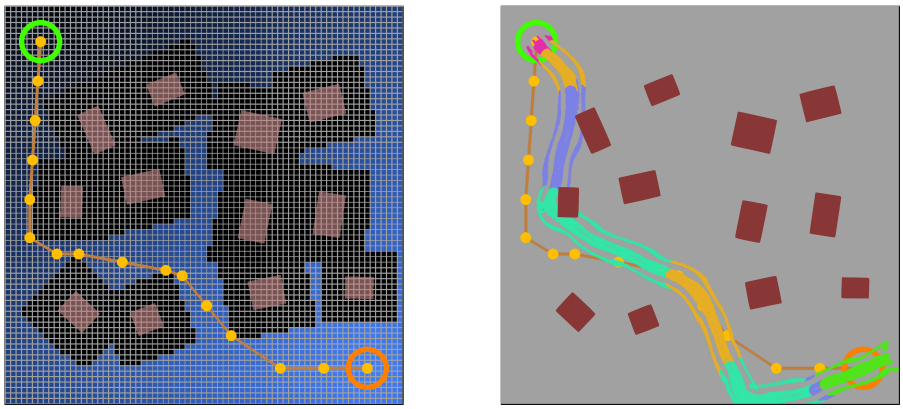


Figure 1.2: Two consequences of decoupled motion search: (left) a trajectory generated using conservative collision assumption, where red areas are obstacles and black areas are the offset to keep the trajectory away from obstacles; and (right) captured motions fail to fit on the trajectory, which result in an animation that penetrates into the obstacle.

generate actual full-body animation. This is commonly addressed as a motion concatenation problem, the aim of which is to find the optimal motion sequence that traverses this path. Many researchers tackle this by using a graph-structured motion database to support querying a series of transitions to obtain a feasible motion sequence (Kovar et al., 2002a; Lee et al., 2002; Safonova and Hodgins, 2007; Lo and Zwicker, 2010).

Whilst this decoupled approach is able to generate animations that appear to avoid the obstacles and reach the goal, such implementations have two consequences: first, to avoid collisions, we must conservatively estimate how far the trajectory should stay away from the obstacle since we cannot determine the exact body movements. We might thus inadvertently exclude shorter paths that allow the virtual human to pass the obstacle more closely. Secondly, since the amount of motion we can capture is finite, the freedom of the mobility of the simplified virtual human is much larger than that of the captured motions combined. Therefore, we might not be able to find exact motions to fit the shape of the entire trajectory and are forced to sacrifice the smoothness of the animation. Figure 1.2 shows an example of the results generated using the decoupled motion search. These drawbacks become more severe when obstacles are dense, causing erratic paths and frequent behaviour-changing animations.

At a different scale, the creation of crowd navigation animation is also completed in two similar stages. In the first stage, the trajectory of each individual is retrieved through simulating crowd interactions where individuals are driven heading towards the goals while avoiding each other. Individual animations are then created by fitting motions to follow these trajectories in the second stage. Most studies are interested in the simulation of the crowd dynamics in the first stage and assume locomotion animation can be generated accordingly to fit their trajectories.

One classical problem is to find global paths for many individuals while taking into account the collision avoidance between them. Such a problem is difficult because once individuals converge, they begin to interact with each other, and as a result their movements become interdependent. One can at-

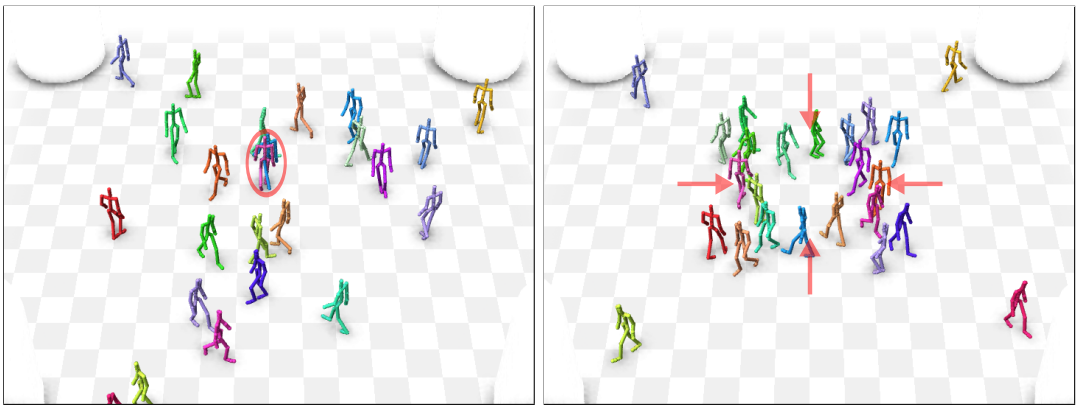


Figure 1.3: Two common anomalous behaviours caused by local navigation models: (left) dispersed crowd formations and inter-penetration between individuals due to the failure of balancing contradictory steering forces; and (right) frequent course and pace changes among the crowd due to the congestions caused by local collision-avoidance strategies. Please refer to the video in Appendix A.2.

tempt to prioritise individuals and plan their paths accordingly, but this becomes infeasible when many of their paths are intertwined. One common way of tackling this problem is to separate interpersonal collision avoidance from global path planning as secondary local navigation that reacts to impending collisions. This is achieved by individually planning the reactions using the local information such as the positions and velocities of the neighbours in the proximity. At each simulation step, these conditions are summarised to determine an action to steer the individual away from colliding into others (Reynolds, 1987; Helbing and Molnar, 1995). However, it can be difficult to balance these contradictory actions and as a result the movement of the crowd becomes very unstable, causing scattered crowd formations and other anomalies. More sophisticated collision-avoidance strategies have been proposed to address this problem. For instance, velocity obstacles (van den Berg et al., 2008a,b) are utilised to improve the prediction of the neighbours' movements. Real-world pedestrian interactions (Pettré et al., 2009) and synthetic visions (Ondřej et al., 2010) are also applied to generate more human-like collision avoidance behaviours. The overall problem of local navigation is that it can only rely on the instantaneous information and the current condition. The result for this is frequent course and pace changes due to incorrect anticipation of collisions. Figure 1.3 demonstrates two crowd animations created using different local navigation collision-avoidance models.

1.2 Contribution

In this thesis, I demonstrate that the *accuracy of collision avoidance* is critical to the creation of plausible navigation animations. The aim of this research is to minimise anomalous behaviours caused by current collision avoidance methods. The overall hypothesis of this thesis is stated as follows:

> *increasing the accuracy of collision avoidance will reduce the anomalous collision-avoidance behaviours, such as making excessive course and pace changes, in character*

navigation, and therefore the resulting animation will be more plausible.

My first observation is that simplifying collision models and approximating motions onto the trajectories are in fact causing anomalous behaviours when avoiding obstacles. The reason for this is that in doing such simplification and approximation, the assumption of collision avoidance must be conservative, which results in a travel path that stays unnecessarily farther away from the obstacles. Furthermore, in order to follow the path as close as possible, one might need to fit more pieces of shorter motions along the trajectory, causing frequent behavioural changes that would sacrifice the smoothness of the animation. As a result, the animation might not be plausible as it is unlikely a real human would behave in an analogous situation. My belief is that both of these problems result from *decoupling motion-planning from motion-synthesis*. Therefore, I argue that to provide plausible navigation animations, the accuracy of collision detection and the continuous and consistent reaction to the obstacles are crucial. My proposal is thus to solve motion planning and motion synthesis simultaneously. The notion to this combination is to support *full-body collision detection in motion concatenation optimisation* such that we are able to find an optimal motion sequence that can travel through obstacles more closely with minimal course and pace changes. Hence, my first contribution in this research is

a joint optimisation technique that extends graph-structured motion database queries to allow full-body collision detection as well as goal-directed motion retrieval in the search of optimal motion sequences for character navigation animations.

My second observation is that when avoiding collisions in a moving crowd, local collision-avoidance models can show instability and fail to react to the congested situation ahead. This is due to the inaccurate collision avoidance caused by the interdependency between near-by individuals when they are avoiding each other. Consequently they begin to impede each other's way and gradually their movements can become very slow and even grind to a halt. The result for this is anomalous crowd interactions and formations that appear to lack anticipation to the collision. This in turn causes erratic travel paths and frequent change of behaviours. Thus, I believe that to create plausible crowd navigation, the spatio-temporal relationships as well as the close interactions between individuals are crucial. To address this, I propose to incrementally identify local incidents where the global paths of individuals interfere and plan their close interactions. The intuition behind this approach is that by planning close interactions with accurate collision avoidance, we are able to minimise the anomalies caused by local interactions so as to reduce the anomalous behaviours for the entire crowd animation. Therefore, the contribution of the second part of the research is

an incremental crowd navigation planning technique that chronologically resolves the interferences between individual animations by planning their close interactions for collision-free and anomaly-free crowd animations.

The aim of this research is to reduce anomalies in character navigation animations. Thus, the evaluation for the techniques presented in this thesis is based on measuring the anomalous collision-avoidance behaviours in the resulting animations. These anomalous behaviours are caused by making

excessive course and pace changes to avoid collisions. They are measured by the travel distance and travel time and the frequency of switching between motions and the results are compared between different methods discussed in Chapter 3 and 4. A more comprehensive benchmarking approach is described in Chapter 5, where a set of plausibility metrics are proposed to analyse the curvilinear motions of individual animations as well as the spatio-temporal correlations between their movements. By evaluating the biomechanical energies, rate of change of accelerations, interpersonal clearances, and the synchronisation between individuals' movements, these metrics are able to highlight anomalous behaviours in navigation animations. The contribution of the final part of the research is

> *a quantitative approach that evaluates the plausibility of navigation animations by measuring their biomechanical efforts, behaviour change rates, interpersonal clearances, and the group synchrony of the crowd movements to highlight their potential anomalous behaviours.*

1.3 Overview

The remainder of this thesis is structured as follows:

In Chapter 2, I review and critique previous work on motion planning and motion synthesis on computer-generated human motions. I explore the path finding techniques on character navigation and crowd simulation. I focus on example-based motion capture blending methods, specifically graph-structured motion database.

In Chapter 3, I present the joint optimisation motion search technique for constructing navigation animations. My technique utilises a graph-structured motion database query to support full-body collision detection in motion concatenation, so it can generate anomaly-free and artefact-free animations that travel through dense obstacles in a close distance.

In Chapter 4, I introduce a navigation planning framework that unifies goal navigation and interpersonal collision avoidance for multiple characters. This approach localise the interferences between individual paths and re-plan their close interactions in order to minimise anomalies caused by interpersonal collision avoidance.

In Chapter 5, I discuss common anomalous collision-avoidance behaviours and propose a set of quantitative metrics to benchmark the plausibility of computer-generated navigation animations. These plausibility metrics provide a side-by-side comparison by directly analysing the resulting animations generated by different methods including the ones presented in this thesis.

Finally, in Chapter 6, I summarise the techniques and results presented in this research and conclude this research with notions to future work.

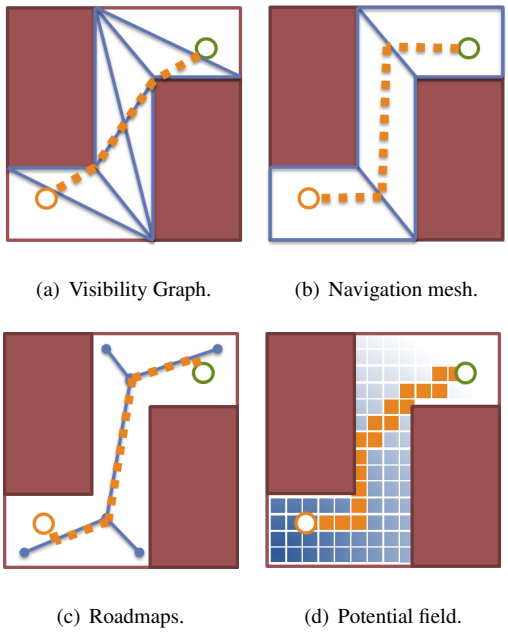
Chapter 2

Related Work

This chapter concerns previous studies on the navigation and collision avoidance of computer-generated human motions. I first discuss motion planning techniques for character animation in Section 2.1. I then describe motion synthesis techniques that utilise motion capture data to generate human animations in Section 2.2. Section 2.3 describes various techniques that generate human navigation animations with and without using path finding techniques. The navigation and collision avoidance of crowd animation are discussed in Section 2.4. The evaluation of synthesised human motion and crowd animation are discussed in Section 2.5.

2.1 Humanoid-character Motion Planning

The purpose of motion planning is to compute a series of collision-free motions to move one subject from one location to another. In robotics, this is formulated as a problem of defining a "configuration space" $\mathcal{C}$ which describes possible states the subject can attain in the free space $\mathcal{C}_{\text{free}}$. We can then draw a curve in $\mathcal{C}_{\text{free}}$ to represent a series of change of configurations between two corresponding locations in the environment in the two dimensional (2-D) task space. Let us consider a simple ground vehicle-like robot which can move around and rotate about to avoid obstacles on a 2-D plane. The configuration space for this vehicle is a three dimensional (3-D) continuous space which comprises x and z position and y-axis rotation. Theoretically, we can unfold the 2-D or 3-D environment into a configuration space with any dimensionality to obtain $\mathcal{C}_{\text{free}}$. For simpler 2-D polygonal obstacles, we might be able to compute their manifold boundaries in the 3-D configuration space to eliminate inaccessible configurations for the ground vehicle. Unfortunately, this is not the case in higher dimensional configuration spaces. To precisely compute $\mathcal{C}_{\text{free}}$ for high-dimensioned subjects such as a human-like character is a hard problem, and is even more so when the environment is complicated. It has been attempted to plan the collision avoidance and coordination between both hands (Koga et al., 1994). To date, there is no efficient implementation on planning full-body human motion in navigation. Therefore, researchers and practitioners often simplify the planning of character navigation as a problem of finding a free path for a point-like robot in the 2-D configuration space, where the boundaries of complex obstacles are easier to compute.



(a) Visibility Graph. (b) Navigation mesh.

(c) Roadmaps. (d) Potential field.

Figure 2.1: Free-space representations: (a) visibility graphs build a network using vertexes of the obstacle. (b) Navigation meshes decompose the free space into convex polygons. (c) Roadmaps are constructed to connect points distributed in the free space. (d) Potential fields are computed to draw a subject into the goal.

One common way of representing 2-D $\mathcal{C}_{\text{free}}$ is to construct networks that describe the interconnectivity of free regions in the environment. Then, a graph structure is built upon such networks to enable graph search algorithms such as the Dijkstra's algorithm or A* algorithm to search for a path that traverses through a sequence of edges or cells connecting two points in the free space. For instance, visibility graphs (Lozano-Pérez and Wesley, 1979) and roadmaps (Kavraki et al., 1996; Choset and Burdick, 2000; Geraerts and Overmars, 2002; Nieuwenhuisen et al., 2007) are utilised to construct free-space network representations to support path finding. A similar free-space representation is also devised to plan camera navigation in a guided tour (Gran et al., 2004). Another approach to acquire free paths is to model obstacles and the goal as repulsive and attractive force fields respectively to describe the kinematic "potential" across the space. Known as the potential-field path finding method (Barraquand et al., 1991), the path from one location to the goal is formulated as a stream cascading down to the low-potential valley towards the goal where the potential is lowest. These $\mathcal{C}_{\text{free}}$ representations are in general very effective in 2-D path finding problems in static environments, but they might not be ideal for avoiding moving obstacles as they require to rebuild the free-space representation whenever the environment is changed. In my single-character navigation study in Chapter 3, I compare my results against the paths generated by the potential field path finding method, which will be discussed in Section 3.3.2.

In computer games, similar free-space representations are applied to find collision-free paths in order to navigate in-game characters. Modern game engines such as Havok Behavior (Havok Inc., 2008) and the Unreal Engine (Epic Games Inc., 2009) provide functionalities for game level designers to specify polygonal objects from the terrain as "navigation meshes" to define accessible free regions in the vir-

tual environment. A graph representation is then constructed to represent the network of free polygons to facilitate path finding in game navigation. Figure 2.1 illustrates four types of free-space representations discussed in this section.

One important issue when reducing character navigation to a 2-D path finding problem is to take into account the minimum clearance for the virtual character to avoid penetrating into obstacles. For a 2-D point robot, the free-space representation $\mathcal{C}_{\text{free}}$ is equivalent to the free space of the virtual environment in the Euclidean space. However, a path found in $\mathcal{C}_{\text{free}}$ for a point robot does not consider the clearance required by a human-like character to pass through. Some techniques use Voronoi diagrams to construct roadmaps so that their free-space networks are always halfway between obstacles (Choset and Burdick, 2000; Nieuwenhuisen et al., 2007), but this approach does not guarantee the clearance for non-point robots and often cause longer journeys to navigate the scene. One way to address this is to compensate the width of the robot in $\mathcal{C}_{\text{free}}$ by thickening the boundary of the obstacle for the distance between the centre of the robot to its sides (Lozano-Pérez and Wesley, 1979). The result is a smaller $\mathcal{C}_{\text{free}}$ of which any free configuration is at least half the size of the robot away from the obstacle. For a rigid-body robot, we might be able to compute the constant widths for each side to expand the obstacles accordingly. For a human-like character, to define the constantly-changing size and shape of the locomotion animation is difficult. One common workaround is to simplify the character as a disk-like robot with a constant radius that covers the boundary of the entire body (Kuffner, 1998; Pettre et al., 2002). More recently, a Delaunay triangulation-based technique is applied to allow path finding for disk robots with arbitrary sizes in complex environments (Kallmann, 2010). The main concern for simplifying the boundary of a character as a disk is the discrepancy between collision accuracy and path optimality. One can conservatively increase the radius to create a larger cushioning space to the obstacle, but in doing so we deliberately reduce the free-space and will thus inevitably increase the journey between two locations. I will demonstrate this problem in the examples in Section 3.4.

Once a free path is acquired, the generation of locomotion is usually addressed as a local navigation problem to steer the human subject to traverse this path. Early human locomotion synthesis studies can be broadly categorised into the kinematic approach (Boulic et al., 1990; Ko and Badler, 1996; Kalisiak and van de Panne, 2001) and the dynamic approach (Bruderlin and Calvert, 1989; Raibert and Hodgins, 1991; Hodgins et al., 1995; Laszlo et al., 1996). The kinematic approach generates human locomotions mainly by solving the inverse kinematics of the lower limbs to reproduce the postures of the phases of human gaits. The computation of kinematic models are in general very fast, but they often fail to capture the physical correctness of the timing and balance observed in human-body movements. Dynamic models simulate human locomotions by applying loop control feedback mechanisms such as the proportional-integral-derivative controllers (PID controllers) to activate the foot strike as well as to balance the lower body for each gait. Joints are modelled using mechanical mechanisms such as springs and pendulums to recreate the physical reactions so that the locomotion animations look more physically-correct. Initially, controlling such interconnected physics-driven joints requires complex calculations and is thus very computationally intensive and unstable. Therefore, early dynamic models could only support forward-

backward locomotions using a lateral planar human model that supports only 1-D hinge joints. More recently, various balance techniques and control policies are studied to stabilise the bipedal movements on uneven terrains (Yin et al., 2007; da Silva et al., 2008) and to support the balance of turning motions using 3-D human models (Muico et al., 2009; Coros et al., 2009; Mordatch et al., 2010). To design physics-based motion controllers that can offer the versatility and agility of natural human locomotion is an ongoing challenging research topic. Whilst the results of these motion controllers are promising, they are not capable of planning close interactions with the obstacle and can only rely on the user input or require a higher-level planner to steer away from impending collisions.

2.2 Graph-structured Motion Database

Concatenating captured human motions is by far the most common approach to synthesise full-body human animation in both research and practise. As motion capture technology becomes more commonly available, one can record a wide range of full-body movements of the performance from a human subject in great detail. Captured motions are arranged as shorter clips which are connected using a graph structure to represent the transitions between these clips. By traversing this graph, we can retrieve a series of clips and stitch them together using short blends at the transitions to create a new animation to accomplish a certain task. For example, one can concatenate a series of locomotion clips, such as ”left turn”, ”right turn”, and ”walk straight”, to generate an animation that navigates through obstacles in a virtual environment. Whilst individual captured motions will look realistic, blending dissimilar clips can compromise the smoothness and the naturalness of the human motion, making new animations look implausible. Therefore, the first challenge of generating plausible blended animations is to identify good transitions between motions.

In game production, blending motion capture data is a common practise for synthesising human animations. Mizuguchi et al. (2001) describe a framework that is used to assist game designers to create transitions for animating characters using captured motions. In their system, the user defines transitions by specifying a tree structure of which the root node is a motion clip that represents a particular action of the character, while the leaf nodes are available transitions to other clips, each of which is again the root node of another tree. The user will then modify the blend range of each transition and examine the blended animation back and forth until he or she is satisfied with the result. Despite these systems requiring extensive user intervention to choreograph motion transitions, such motion blending techniques, also known as blend trees, remains very popular among modern game engines (Havok Inc., 2008; Epic Games Inc., 2009; NaturalMotion Ltd., 2009).

To reduce the burden of manual inputs, researchers have exploited automatic ways of detecting transitions from captured motion data. Early studies on the detection of transitions began with statistical models that estimated the transition probabilities of the temporal sequences observed from captured motion data. Stochastic processes, such as Markov chains and hidden Markov models, were used to establish the state-transition probabilities between poses or clips (Bowden, 2000; Brand and Hertzmann,

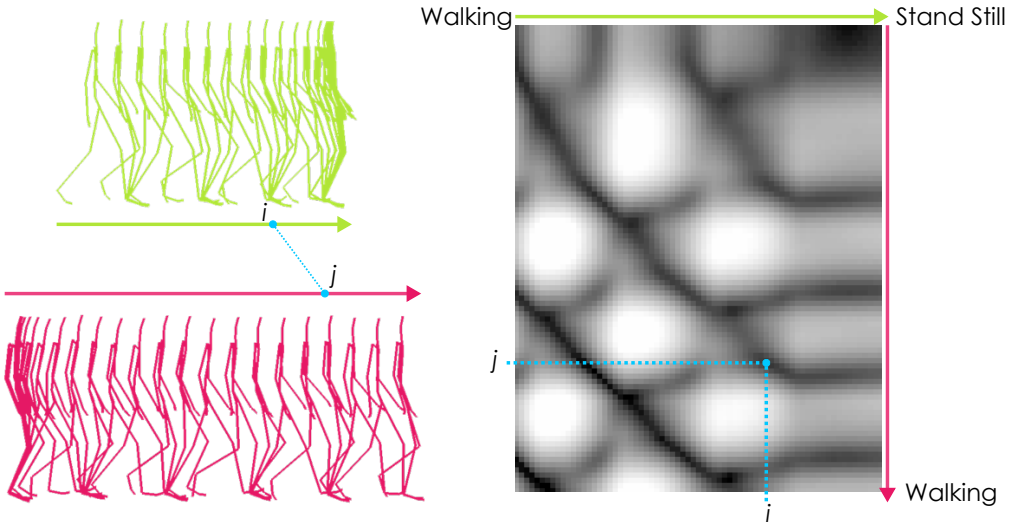


Figure 2.2: A joint position-based distance matrix computed between two motions. Dark regions indicate poses that are numerically similar.

2000; Tanco and Hilton, 2000). These methods avoid poor transitions by only accepting those with high probabilities. To synthesise continuous animations, they either explicitly or implicitly employed graph-like structures to describe the flow of transitions between motions. Thus, they are considered as the earliest examples of graph-structured motion database. Estimating transition probabilities indeed enables us to find candidate transitions occurred in the captured data, but it cannot recognise transitions that have not been observed before.

To maximise the connectivities between motions, new methods to measure numerical pose-to-pose similarity are investigated. Aroused by the seamless video synthesis technique by Schödl et al. (2000), various frame distance metrics are utilised to discover transitions by measuring the differences between poses based on joint rotations (Lee et al., 2002; Pullen and Bregler, 2002) or joint positions (Kovar et al., 2002a; Arikan and Forsyth, 2002). Whilst joint rotation-based distance metrics are considered to be easier to implement, joint position-based methods have the advantage of matching poses by the closeness of their foot contacts and hand-grasping positions, which reduces common motion-blending artifacts such as foot skating. Once a distance metric is defined, we can calculate a distance matrix to represent the cost of each transition between the i^{th} frame of one motion and the j^{th} frame of another. Figure 2.2 shows an example of a distance matrix calculated between two walking motions using a joint position-based metric. We can then accept those transitions where costs are under a certain threshold as transitions between these motions. By pair-wisely discovering the transitions between clips in the captured motion data, we are able to build a directed graph to describe the flow of the entire motion database. Any path obtained through the traversal of this graph represents a sequence of edges that describe a series of transitions, and the clips between the edge sequence can then be stitched as a continuous and natural-looking animation. Such graph-structured motion transitions based on their numerical similarity are commonly referred as *motion graphs*.

One of the most common usages of motion graphs is to retrieve a sequence of clips to accomplish

a specific request. In navigation animations, such requests can be to query the motion graph for an animation that travels across the space to reach a goal location. This is often formulated as a cost minimisation problem: by defining a *cost function* to evaluate how well a solution achieves the goal and a *search algorithm* to traverse the graph for a solution that minimises the cost. The cost function is defined by specifying a set of constraints, such as the goal location and the way-points to travel through, that a solution must satisfy. The cost of the solution is then calculated by measuring the deviation between the trajectory of the animation and the constraints. One important issue is to minimise the number of transitions required to achieve the goal while maintain the smoothness of the animation. This is addressed by including the costs calculated using the pose similarity metric for each transition in the cost function to penalise a solution that requires frequent transitions.

Searching for optimal solutions with motion graphs is known to be a hard problem. This is because there often exist many similar poses in the motion database, especially those consist of human locomotions, and thus one can always find a transition to continue traversing the graph. This results in a cyclic graph where exhaustive search is impossible. Initially this is tackled by applying local search methods such as the branch-and-bound algorithm (Kovar et al., 2002a) or a bounded-depth best-first search (Lee et al., 2002) to find solutions that satisfy the cost function. But these algorithms become infeasible as the size of the search space increases exponentially when the size of the graph grows. As shown in Figure 2.2, once we have found a transition between frame i and frame j, their adjacent poses would also be very similar and together they form a set of transition cluster. To reduce the graph to a more manageable size, graph pruning techniques such as finding strongly-connected components are applied to drop the edges that lead to dead ends (Kovar et al., 2002a; Lee et al., 2002). Further graph culling can be done by removing the transitions of mismatched phases in human gaits, e.g. transitions with mismatched supporting foot in walking motions, to reduce the number of the edges in the graph (Lee et al., 2002; Safonova and Hodgins, 2007). However, the trade-off between the size and the connectivity of the graph is critical to the effectiveness of the motion graph, an issue that is discussed by Beaudoin et al. (2008) and (Zhao et al., 2009).

Another way of querying motion graphs is to apply randomised search. This approach finds feasible solutions by initialising random edge sequences (seeds) and then iteratively making local modifications (mutations) to their subsequences in order to minimise the costs. Stochastic optimisation methods such as Monte Carlo simulation has been applied to generate a series of physically-correct motions that fulfils certain constraints to control rigid-body animations (Chenney and Forsyth, 2000). By exploiting a similar random sampling technique, Arikan and Forsyth (2002) optimise the solutions towards satisfying the constraints by systematically exploring the motion graph for alternative subsequences. Instead of pruning the graph, a vector quantisation technique is applied to summarise each cluster of adjacent transitions into a binary search tree to reduce search complexity. Throughout the optimisation, root-level edges are replaced to modify the composition of the transitions (hop mutations) while child-level edges are traversed to fine-tune each transition (demote mutations). These mutations are repeated until each candidate is converged to the local optimum where no modifications can be made to improve the solu-

tion. By searching the motion graph with multiple random seeds, this technique is able to choose among several local optimal solutions that fulfil the same goals. The motion database query in this thesis is based on randomised motion graph search. I will discuss this how I implement this approach to support character navigation and collision avoidance in more detail in Section 3.3.1.

2.3 Query-based Character Navigation

To navigate through obstacles with motion graphs, one common approach is to query for an animation that follows a free path. Previously in Section 2.1, we have discussed various free-space representations and the path finding techniques used to traversing them for free paths. To query an animation that follows a free path, local search is first applied to incrementally approximate the curvature along the path to obtain a navigation animation (Kovar et al., 2002a; Lee et al., 2002). The quality of approximation can be improved by interpolating motions (Safonova and Hodgins, 2007) while the performance can be sped up by using a bidirectional A* search (Lo and Zwicker, 2010). Path following can also be achieved via randomised search (Arikan and Forsyth, 2002) by laying as many way-points as necessary along the path, a technique that will be discussed in Section 3.1. The main concern of this approach is that in order to avoid the obstacle, one needs to follow the path as closely as possible. As a result, the animation does not always appear to be plausible as it often requires more smaller motion clips in order to fit the path more tightly. This is because the information of obstacles is no longer available at this stage and one can inadvertently collide into an obstacle if the queried animation deviates from the path. In other words, the motion query is over-constrained by fitting the path as one might be able to find an alternative solution that chooses fewer longer motions to avoid the obstacle. I demonstrate this problem by measuring the transition frequency of path following in Section 3.4.

An alternative to path following is to describe the accessible free space using motion graphs and then to traverse this graph for an animation to navigation the environment. One simple implementation is to construct a motion graph with captured motion stored in the task space using the coordinate system where motions are recorded. For example, Lee et al. (2002) split a long playground motion capture data into several motion clips that represent different choices at several certain transition points, such as climbing over the frame or ducking under the frame. The user is then prompted at each transition point to make a decision so that the system can play back the corresponding motion clip to the next transition point. A motion graph constructed in the task space is known to be “fixed-coordinate” as the motion clips are preserved using the same coordinate system as they were captured. Fixed-coordinate motion graphs are ideal if the human-environment interactions are critical, such as in the case of playground animations where the contacts of the climbing frame are noticeable. For larger environments they might not be applicable as it can be difficult to capture and store all possible motions and their transitions.

A more general approach to fixed-coordinate motion graphs is to extend existing free-space representations (see Section 2.1) with interconnected sub-graphs extracted from motion graphs. This approach often requires an off-line graph search stage to embed appropriate sub-graph paths into the free space of

the environment. For instance, by fitting the footsteps of the animations queried from the motion graph on a probabilistic roadmap, Choi et al. (2003) can produce long navigation animations that traverse a selection of locations on the terrain. However, the navigation of this scene is limited to these locations. To allow the exploration of the entire free space, Reitsma and Pollard (2004) and Safonova and Hodgins (2007) both unroll pieces of sub-motion graphs to enumerate viable navigation animations between any two locations in the grid space of the environment. In doing so they can traverse the sub-graph paths embedded between the cells for navigation animations that travel across the scene. One major limitation for these free-space representation extensions is that they cannot deal with moving obstacles as it requires to re-construct the free-space representation whenever the environment is changed. Another drawback is that the viewer may find the navigation animations always look identical when traversing the same part of the scene.

One way of avoiding moving obstacles with motion graphs is to associate the status of an impending obstacle with an action to avoid the collision. This is often formulated as pre-computing the mappings, known as the *control policies*, between the configuration of a certain state and the corresponding motion sequences from the motion graph, such that the reaction animation can then be retrieved accordingly at runtime. One early example to facilitate such character motion control policies is done by Lee and Lee (2004), where a set of boxing motion clips is associated with a state configuration to allow interactive control of the punching and footwork actions. The control of punching motion, for example, is determined by the configuration in a state space indicating the relative positions of the target to the boxer. At run-time, a state configuration is then used to look up for an animation that appears to be punching at the target. By defining a footwork and a punch control policies, this technique is able to generate dynamic boxing animation of two boxers approaching and punching each other interactively. Whilst they can also generate secondary animations to simulate the reaction when a boxer is punched by the other, they do not consider how to avoid an attack or the collision between the boxers.

To realise collision avoidance in the control policies, Lau and Kuffner (2006) pre-compute a set of short paths that can be incrementally expanded to longer ones, called pre-computed search trees, in order to find a collision-free path to the goal. Each of these trees is associated with a discrete state, which is the control policy of a selection of similar motions with slight variations such as moving towards different directions. The transition between these states is triggered at search time by a finite state machine so that the paths are expanded from one state to another accordingly. To avoid local optimums in static scenes, a coarse-level potential-field path finding can also be applied to guide the expansion of the search trees. At each state, instead of using a state-variable lookup table, a path is chosen by excluding those branches obstructed by the obstacles and then choosing the shortest one towards the goal. Pre-computed search trees can also be applied to support collision avoidance in dynamic environments. This is done by interactively choosing a path that avoids the current position of the obstacle on a per-state basis. Therefore it can only plan collision avoidance for moving obstacle over a short period of time. Furthermore, since these trees are constructed using the ground trajectories of the root joint, the collision detection of this technique needs to be very conservative and hence is inaccurate. My joint motion search

also applies collision detection to optimise a global path towards collision-free. It is a off-line method that supports full-body collision detection so the collision avoidance of the resulting animation is more accurate.

Another approach to support obstacle avoidance in control policies is to apply reinforcement learning techniques. The key benefit of constructing control polices using reinforcement learning is that they can plan long-term animations to avoid obstacles. Instead of manually selecting a set of short motion clips to construct each state space, this approach learns a motion sequence for a given state configuration by querying a motion graph with a reward function. By maximising the long-term reward of each motion sequence, one can build a near-optimal control policy to avoid obstacle situations learnt in the state space. Treuille et al. (2007) first describe long-term planning control policies learnt using a delayed rewards model. Their obstacle-avoidance control policy can avoid one simple disk-like moving obstacle. Lo and Zwicker (2008) apply another reward strategy that can directly analyse the task space so that they can achieve more precise motion control such as grasping an object. This technique is extended to support vision perception to allow avoiding multiple obstacles (Lo et al., 2012). Lee et al. (2009) further extend Treuille et al.and Lo and Zwicker's models to support a finer resolution collision detection so that they can avoid dynamic obstacles such as a revolving door. However, the fundamental limitation of reinforcement learning control policies is that the state space often grows rapidly when the size of the motion database and the complexity of the obstacle increase. Thus they are considered as local navigation controllers and can only deal with very limited numbers of obstacles.

2.4 Distributed and Data-driven Crowds

The simulation of crowds has attracted a lot of interest in the research community. Within computer graphics interest in the topic was sparked by the work of Reynolds (1987) on simulation of flocking behaviour based on computation of local interactions between each individual. Another seminal work from physics is the social forces model by Helbing and Molnar (1995), which is somewhat similar to Reynold's model, but based on pedestrian behaviour. These two algorithms are the classic examples of *local models* which simulate crowd behaviours based on certain local control policies of each individual. In local models, each individual is driven by the urge to approach its target and to flee from hazards such as colliding into obstacles and other individuals. Such behaviours are often realised as summing several attractive and repulsive forces acting upon each individual to negotiate its movement over the course of time. Collectively, local interactions can form interesting emergent behaviours. For instance, attractive and repulsive forces are applied among individuals to enable cooperative and adversarial interactions such as leader-following behaviour and pursuit-evasion interaction (Reynolds, 1999). In the case of crowd navigation, these force fields are utilised to attract individuals towards their goals and to repel each other to avoid collisions. From motion-planning point of view, this approach can be seen as a distributed planning via a multi-agent system, where individuals constantly refine their plans to negotiate or coordinate with others while achieving their own goals.

One problem with classic local models is that they do not consider spatial-temporal reasoning to avoid collisions but rely only on repulsive forces to repel each other. Thus they can show instability in crowded situations and appear to lack human-like anticipation to the collision. Many extensions have been proposed to address this problem. For example, a spring-like structure is maintained to separate individuals apart while stabilising their formation (Goldenstein et al., 2001). Rule-based environment perception models are defined to react to various obstacle situations to enable autonomous pedestrian behaviours (Shao and Terzopoulos, 2005). "Velocity obstacles" (VOs) are computed for each neighbour to prevent moving into regions that lead to potential collisions (van den Berg et al., 2008a,b; Guy et al., 2009). Captured motions are stitched and indexed such that collision detection can be done in the Euclidean space (Lau and Kuffner, 2005) or appropriate avoidance motions can be looked up in the parametric space to avoid an obstacle (Treuille et al., 2007). More recently, real-world pedestrian interactions (Pettré et al., 2009) and synthetic visions (Ondřej et al., 2010) are also applied to reproduce human-like collision avoidance behaviours. These local navigation techniques provide alternative strategies to react to obstacles more reasonably. However, the fundamental limitation for local models is that they can easily create deadlocks when many individuals' paths interfere. This is because when individuals begin to converge their movements become interdependent, and thus one can only react to others based on their instantaneous movements until eventually little progress can be made at each timestep. This often results in anomalous behaviours such as frequent pace and course changes, which can be seen in the examples in Section 4.5.

Local models are commonly applied among commercial animation softwares. For example, Massive Software (2009) provides the user a set of building tools to create a fuzzy logic network that is used to steer crowd individuals, called agents, away from the collision. A fuzzy logic network often begins with the input of certain agent perceptions such as the distances and the bearings to other agents. The user is provided with various functions to define the mapping between the perceptions and desired reactions to steer the agent. For instance, one can define the moving speed of an agent is inversely proportional to the distance to the neighbours so that the agent slows down when the proximity is getting crowded. Each of such reactions is associated with a fuzzy threshold and at runtime, the reaction of an agent is triggered by random numbers to determine the combined steering behaviour in respond to the perceptions. To control the crowd movement, the user is allowed to manually create a static flow field to direct the traffic of the crowd. The limitation of this approach is that the flow field cannot respond to the density of the crowd and thus it can still create congested and deadlock situations.

In order to prevent deadlock situations, some form of high-level planning is required to conduct local navigation to react to the congestion ahead. One common strategy is to organise individuals into groups and plan their global paths accordingly using the free-space representations described in Section 2.1. For instance, free-space networks such as navigation graphs (Pettré et al., 2005; Morini et al., 2007) and roadmaps (Kamphuis and Overmars, 2004; Nieuwenhuisen et al., 2007) are applied to choose a global path for a group of near-by individuals that share the same initial and goal locations. These algorithms assume individuals in a group move in a "flock-like" behaviour so that they can estimate the

clearance required by the group to determine whether a path is passable. However, they can still create congestions when two or more groups converge, since each path is planned independently. To take into account the global density of the crowd, a density map (van Toll et al., 2012) is applied to penalise congested paths in order to divert the flow of the crowd. The free space is decomposed into cells and at the runtime the population in each cell is used to determine the moving speed in this region so that paths within each cell are weighed accordingly for an A* path search. One limitation of these algorithms is that the individuals in their crowds have to travel in cohesive groups. Thus they are not applicable if each individual has its distinctive goal, such as the multi-character navigation problem addressed in Chapter 4.

An alternative crowd navigation planning technique is the dynamic potential field (Treuille et al., 2006), which is computed based on the goal locations of the individuals as well as their current positions to control the flow of the crowd. At each timestep, a potential gradient is created for each goal location so that individuals travelling to a common goal can be drawn towards the location of the lowest potential. Each of these goal gradients is then combined with the potential gradient of the current global density of the crowd to disperse individuals away from crowded regions. As a result, one can sometimes observe anomalies where individuals repel each other similar to the anomalous behaviour caused by the repulsive force fields of the local models. This is addressed as an incompressible flow problem where the local density of the crowd is set to be a constant to maintain the minimum clearance between individuals while following the global flow (Narain et al., 2009). This technique can somewhat soften the flow field in reducing the repulsive behaviour. By tracing the flow field formed by these potential fields, one can obtain global paths that appear to avoid congested areas. Crowds driven by flow fields no longer need to move in cohesive groups, although the computation of the potential fields will be more efficient if many of the individuals share the same goal. For situations where the structure of the flow of the crowd is less obvious, we might need to compute one potential field for each individual with its unique goal.

Another interesting crowd animation approach is to model crowd behaviours and interactions from the video recordings of real crowds. This approach applies computer vision-based techniques to analyse the trajectories of the crowd from the video in order to reproduce their behaviours. For example, a flow field is computed based on the crowd trajectories in the video so that it can be used to control the individual movements of the virtual crowd (Courty and Corpetti, 2007). This technique is only suitable when the initial conditions of the real crowd and the virtual crowd are similar. Instead of collectively control crowd motions, Lee et al. (2007) build a steering behaviour model by analysing the formation of groups from labelled crowd trajectories so that individuals can coordinate with their neighbours to portray certain crowd behaviours. By mapping the distribution of the neighbours with corresponding reactions from the video, they can reproduce real-world crowd interactions such as group chatting, queueing, and wandering. They can even apply the behaviours of the ants modelled from video recordings on virtual human crowds. This technique is extended to control the behaviour of larger crowds, so that their formation can be interpolated between different types of crowd interactions (Ju et al., 2010). These data-driven crowd animation techniques view crowd interaction as a crowd formation problem. As long as the formation of

the crowd is maintained no explicit collision avoidance is required. Therefore this approach is not ideal for situations where the crowd does not have an obvious formation.

So far most of these crowd navigation techniques discussed in this section are centred on the planning of the trajectories of the crowd movements. In other words, they are only interested in simulating coarse-level crowd behaviours. In order to generate animation that appears to be a human crowd, human locomotion animations must be generated to follow the trajectories of individuals. One simple implementation is to place human walk cycle motion clips onto the trajectory with their forward vectors aligned the tangent of the curvature. This approach can generate locomotion animations on-the-fly. But it does not take into account the nonlinear pelvis movement of the human gaits. As a result, visual artefacts such as foot skating can be easily seen in the resulting animations. Some studies address this as foot planting problem (Kovar et al., 2002b; Ikemoto et al., 2006), but this can create unnatural postures if the direction and speed between the gait and the trajectory is mismatched in the first place. Another off-line solution to generate locomotion animations is to query the motion graph for animations to follow these trajectories. As discussed in Section 2.3, although it is possible to generate artefact-free animations using motion graphs, the results are not always optimal as we might not be able to fit motions on the trajectory. I will demonstrate this problem in multi-character navigation experiments in Section 4.5.

One way to directly generate crowd animations with human locomotion is using group motion editing. This approach arranges multiple individual locomotion animations as a group of characters moving together and modifies their paths as a whole. This approach often relies on a spring-like structure (Goldenstein et al., 2001) to maintain the interpersonal clearance as well as the formation of the group. For example, a motion time-warping technique is applied to modify the paths of individual animations so that the formation of the group is maintained (Kwon et al., 2008). Individual animations can be generated using agent-based motion models while the formation of the crowd can be guided with user-specified curves (Oshita and Ogiwara, 2009). This idea can be extended to support obstacle avoidance (Henry et al., 2012). Like crowd path planning techniques, crowds control using this approach must travel in a group. Also, the spring-like structure does not allow the change of relative position between individuals, which can be inapplicable for many crowd navigations.

An alternative approach to create crowd animation is to directly assemble blocks of pre-computed navigation animations. This is often done by fitting motions to a 2-D rectilinear grid so that a motion path can be decomposed into segments within the cells, known as motion patches. By establishing the transition between each patch, one can piece together new animations moving from cell to cell. Lee et al. (2006) construct a set of single person-sized motion patches that support various environment constraints so that a cell can be associated with a certain type of motion patch. For instance, when moving into a cell of a chair, a sitting motion patch is chosen to animate the character accordingly. Interpersonal collision avoidance can also be done by only allowing one character to occupy a cell at a time. Thus, during local search, cells that are occupied will be excluded from the search space to avoid potential collisions. This idea of motion patches has been extended to support multiple character interactions in a larger cell, called the crowd patch (Yersin et al., 2009). Each crowd patch is pre-computed so that individuals can appear

to be interacting with or avoiding each other. Thus, they do not require run-time collision avoidance as motions embedded in the same patch are guaranteed to be collision-free. The main drawback of the patch-based approach is that the choices of a patch can be very limited as each cell is constrained by the types and behaviour of the adjacent cells because the individual must be able to move from one cell to another. Furthermore, since individuals can only change their navigation on a per-cell basis, the behaviour of the resulting animations do not always look plausible as they cannot change their navigation freely.

Motion patches can be used as spatio-temporal constraints to compose crowd animations. By specifying pre-defined character interactions as constraints, this approach optimises individual animations by sequencing and modifying transitioning motions between these motion patches. Therefore it is able to generate continuous animations without character interactions being confined to a rectilinear grid. Liu et al. (2006) describe a multi-character motion composition framework to sequence a series of time-warped motions that join together user-specified character interactions where one character is dodging the tackle from the other. Their motion time-warping is guided through a space-time optimisation to maintain the physical correctness. The user can decide whether the tackle is successful such that the ending animations for both characters are chosen accordingly. Shum et al. (2008) pair up character interactions as motion patches to form a patch-level motion graph so that crowd interactions can be generated by traversing this graph. Kim et al. (2009) also propose a multi-character animation editing technique based on motion time-warping, but they require the user to specify the temporal sequences and the motion paths of the characters in order to generate their interaction animations. This technique is extended by Kim et al. (2012), where the interactions are modelled a set of deformable motion patches so that they can be seamlessly assembled as crowd interactions. While motion patch methods avoid collision in pre-computation, this technique further supports inter-patch collision avoidance using a bounding volume collision detection. However, the main interest of these techniques is to compose crowd animations under user-specified constraints rather than crowd navigation. My crowd navigation planning technique automatically identifies local interactions to resolve collisions with any user interventions.

2.5 Synthetic Human Motion Evaluation

The evaluation of synthesised human motions has drawn considerable attention in the computer graphics community. Early studies begin with the analysis of the physics correctness of human motions such as the ballistic trajectory and the angular momentum of a jumping motion (Reitsma and Pollard, 2003; Safonova and Hodgins, 2005). Such metrics are only applicable with animations that are driven by physics simulation-based methods. To evaluate the naturalness of interpolated motion capture data, supervised classifiers such as support vector machines are applied to examine partially grafted motions to be added into a motion graph (Ikemoto and Forsyth, 2004). With a similar classifier, pairs of foot-contact-aligned motion clips are associated to be blended into new transitions that would still look natural (Ikemoto et al., 2007). To determine the blend strategy of the transition, cross validation and user studies

are conducted to find the optimal blend range and weight for different motions (Wang and Bodenheimer, 2003, 2004). By feeding a large motion database as a prior, Gaussian mixture models, hidden Markov models, and a linear dynamic system learning technique are used to highlight anomalies in synthetic human animations (Ren et al., 2005). These methods focus on the visual qualities of the animation, namely the continuity and the phase synchrony of the synthesised motions.

To evaluate the behaviour of a navigation animation, Reitsma and Pollard (2004) define metrics that measure the accessibility and the path lengths queried from a motion graph. To determine the quality of a path, they calculate the ratio between the length of the actual path and the linear distance between the initial and goal locations. My plausibility metrics also evaluate an navigation animation by comparing based on a hypothetical path, but I estimate the biomechanical energy required to traverse the path rather than calculating the arc-length of this path.

For crowd animations, one of the main concerns is avoid anomalous patterns in the constituents of the crowd. Studies have shown that humans favour high variation in the appearances of individual crowd members (McDonnell et al., 2008) while the composition of the crowd in a still image is more scene-dependent to human perception (Ennis et al., 2008). To provide sufficient variation in crowd animations, a collection of human motions is evaluated so that they can be deployed more randomly to avoid unintentional formation and synchronous movements (Pražák and O'Sullivan, 2011). To evaluate the behaviour of crowd navigation, real-world motion data has also been exploited. Pettré et al. (2009) compare the reaction to a collision generated using various collision-avoidance techniques to captured pedestrian interactions, but they only consider the immediate reaction to an impending collision. Normally, data-driven behaviour evaluations are case-specific. For example, one can measure the spatial randomness of synthesised crowd movements and compare that to an existing crowd formation (Ju et al., 2010). Recently a likelihood estimation model is applied to measure the similarity between a real-world crowd dataset and a reconstructed, synthesised crowd motion (Guy et al., 2012). These methods are not general as they cannot determine whether the behaviour of a crowd is genuinely plausible without relating it to a close example.

2.6 Summary

In this chapter, I have reviewed various motion planning and motion synthesis techniques concerning navigation animations for computer animated characters. I have also described the state-of-the-art methods that evaluate computer-generated human motions. Both single-character scene navigation and multi-character collision avoidance are covered when discussing these techniques.

The planning of navigation animation in previous studies is often decoupled into path finding and path following, so as to reduce the search space and to make use of motion capture data, respectively. To guarantee the animation to be collision-free, one must choose a free path that stays away from obstacles to allow a certain clearance for the character to pass through. Fitting motions on a path can be difficult as the search space increases exponentially with the size of the motion database. Another disadvantage is

that this decoupled approach does not deal with moving obstacles. I view character navigation as a joint optimisation problem, with the combination of collision avoidance and motion concatenation. I directly search the motion database for a series of motions to avoid obstacles. Therefore I am able to plan close interaction using full-body collision detection. Also, the resulting animations are plausible in that they do not contain visual artefacts such as foot skating.

When considering the navigation for multiple characters, local navigation is often used to avoid colliding into each other. This approach can cause regional deadlocks easily and thus requires a high-level planning to control the flow of the crowd. However, current global planners cannot regulate the flow of the crowd effectively if the flow of the crowd does not have obvious structure and cannot arranged into groups. Motion patches are alternatives to create multi-character interactions, but their crowd animations are confined by the combinations available in the building blocks. My crowd navigation technique unifies the planning of goal navigation and interpersonal collision avoidance. Therefore I am able to globally control the flow of the crowd while minimising regional congestions. My interpersonal collision avoidance is done by planning interactions between individuals via motion database queries. Thus the resulting animations are plausible in that the individuals can avoid each other closely without anomalies such as repulsive movements or frequent change of motions.

Finally, previous evaluation studies centred on the visual quality of the synthesised human motions and the variety of the crowd animations. Path lengths can be considered as an indicator to measure the effectiveness of a navigation animation. Unintentional formation and synchronised movements are regarded as anomalous behaviours in crowd animations. Current navigation and collision avoidance evaluation methods require training data for cross validation. To date, there is no general approach to evaluate the plausibility of navigation animations.

Chapter 3

Single-character Navigation

To construct navigation animations it is common to stitch together a series of motions from a human locomotion database. Typically, to avoid obstacles, the motion generation is decoupled into two stages: finding a collision-free path to the goal (motion planning) and then fitting captured motion data to follow this path (motion synthesis). Whilst the generated animations can still look smooth, the results do not always look plausible in that they do not resemble the behaviours a human would respond in analogous real-world situations. In this chapter, I investigate the implausible situations caused by conventional decoupled motion generation and present a new motion query technique that combines the planning of collision avoidance and the synthesis of full-body movements to generate human-like navigation animation. My observation on the conventional approach is that the separation of motion planning and motion synthesis is in fact generating sub-optimal animations, causing detoured and delayed paths when avoiding obstacles. The results can thus be less plausible. I demonstrate that by incorporating collision detection into motion database query, my technique is able to find shorter navigation animations interacting with the environment more naturally.

In this chapter, I will first discuss the methods and major pitfalls of the conventional decoupled motion search, referred to as *path finding and following* in Section 3.1. I will then describe how to enable collision avoidance in my *environment-aware motion graph* technique to achieve motion planning and motion synthesis simultaneously in Section 3.2. The implementations of both methods are described in Section 3.3. The comparison of both methods will be evaluated by measuring the travelled time and distances as well as the required behavioural changes of their resulting animations using a series of test scenes in Section 3.4.

3.1 Path Finding and Following

Concatenating motion capture data is considered an effective way of reusing a finite set of human motions to compose a broader range of realistic-looking animations. One common practise is to utilise a graph-structured database to store the transitions between similar motions among captured data. Animations are generated by traversing this graph to obtain a series of transitions and then stitching together the motion

clips connected by these transitions. Since these transitions are identified between similar motions, it only requires a short linear blending at the end of two clips to generate smooth long animations. Such database query technique is also known as *motion graphs* (see Section 2.2).

To query a motion graph, one can formulate a set of constraints, such as initial and goal locations and certain waypoints to travel through in-between, and search for a sequence of motion clips that satisfies these constraints. This is often realised by defining a *cost function* to evaluate how well a solution, a motion clip sequence traversed from the graph, fulfils the constraints and a *search algorithm* to systematically explore the graph for a solution that minimises the cost. Let $\mathcal{M}$ be the set of motion clips in the database such that $\mathcal{M} = \{m \mid m \text{ is a clip}\}$. Given a solution $\bar{x}$ traversed from the graph, we obtain a clip sequence connected by the transitions as $\bar{x} = (m_1, m_2, m_3, ... \mid m_i \in \mathcal{M})$. We can derive the position of each frame of $\bar{x}$ by accumulating the location and orientation of the corresponding clips m_i from the initial location. I denote the positions of the poses derived from the accumulated motion clips as a sequence $P = (\rho_1, \rho_2, \rho_3, ...)$ such that ρ_i is the world position of the root (pelvis) joint of i_{th} pose. Let $(\varrho_1, \varrho_2, ..., \varrho_n \mid n \text{ is the number of waypoints})$ be the series of waypoint and q be the goal location. The cost function to determine the deviation between the animation and the constraints is defined as

$$C_{\text{deviation}}(\bar{x}) = \underbrace{\frac{|p-q|}{\Delta}}_{\text{goal}} + \underbrace{\sum_{k=1}^{n} \left(\frac{|\bar{\rho} - \varrho_k|}{\Delta}\right)^{(n-k+2)}}_{\text{waypoints}}, \tag{3.1}$$

where p is the final position of $\bar{x}$ and $\bar{\rho}$ is the position of the nearest frame of $\bar{x}$ to a corresponding waypoint ϱ_k, and is found by

$$\bar{\rho} = \underset{\rho_i \in P}{\arg\min} \, |\rho_i - \varrho_k|, \tag{3.2}$$

where $u < i < n$ and of which u is the corresponding frame index of previous waypoint to prevent degeneration. The exponent $(n - k + 2)$ of the waypoint term in equation 3.1 is used to prioritise waypoints such that $\bar{x}$ travels through them in the correct order. The parameter Δ is a positive non-zero scalar to penalise the deviation so that each $\bar{\rho}$ can stay close to its constraint ϱ_k. Therefore, the aim of the search algorithm is to find an $\bar{x}$ that minimises equation 3.1.

In order to navigate through obstacles, we require a sequence of a combination of various motion clips such as “left turn”, “right turn”, and “move forward” to reach the destination. To generate such navigation animations, a common strategy is to plan a ground trajectory to the goal and then query the motion graph to find the sequence of clips that follow this trajectory accordingly. Figure 3.1 illustrates a schematic view of conventional decoupled motion planning and motion synthesis approach, referred as *path finding and following*. The process begins with the goal specified by the user. Then, at the path-finding stage, a free-space representation of the virtual environment is utilised to obtain a collision-free trajectory to reach the goal. Once a path is found, it is then used as a constraint to traverse a sequence of clips from the motion graph by planting as many waypoints as necessary to confine the animation stay closely to the trajectory in the path-following stage.

Since the path is the sole information exchanged between the two stages, many researchers have attempted to improve motion graphs to approximate motions on the trajectory more closely to facilitate

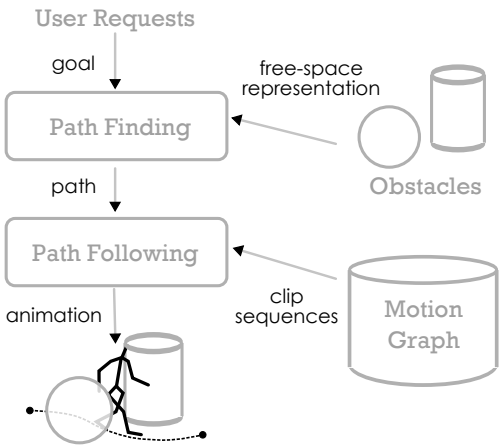


Figure 3.1: Path finding and following.

obstacle avoidance. For example, motion interpolation can be applied to fit motions on an input curve through graph traversals (Safonova and Hodgins, 2007; Lo and Zwicker, 2010) as well as to create more synthetic transitions to increase the connectivity of the graph (Ikemoto et al., 2007; Zhao and Safonova, 2008; Ren et al., 2010). Path finding problems are well-studied in robotics and artificial intelligence. Techniques such as the Dijkstra search or the A* search algorithms are applied to retrieve a shortest path using roadmaps (Kavraki et al., 1998; Choi et al., 2003; Nieuwenhuisen et al., 2007) and discretised free-space representations such as a grid or a set of connected polygons (Barraquand et al., 1991; Kuffner, 1998; Bandi and Thalmann, 1998; Gran et al., 2004; Pettré et al., 2005; Kallmann, 2010). However, the conventional decoupled approach has two fundamental pitfalls:

- **Simplification.** To simplify navigation as a path-finding problem, we must make a conservative assumption about how far to stay away from the obstacle as we have no information of the exact body movements available at this stage. One common practise is to regard the virtual human as a moving circle with a *maximum bounding radius* estimated from the captured motions to determine whether the free space is passable when planning the trajectory. We might thus exclude shorter paths that allow some motions to travel through the obstacle more closely to the goal. This means that the planned path might not be optimal in the first place.

- **Approximation.** As the environment is no longer available in the path-following stage, we are forced to closely approximate the trajectory in order to avoid collisions. One way to achieve this is to deliberately bend motions towards the curvature of the trajectory by interpolation, but we might thus destroy the smoothness and the naturalness of the captured motions. Alternatively, we could fit short pieces of clips on the trajectory using a super-sampled graph to minimise the deviation, but doing such might require more than necessary clips in order to navigate to the goal. This often results in frequent transitions which is considered as signs of anomaly and contradicts the least-effort assumption.

As a result, the simplification of virtual human and the approximation of the trajectory over-

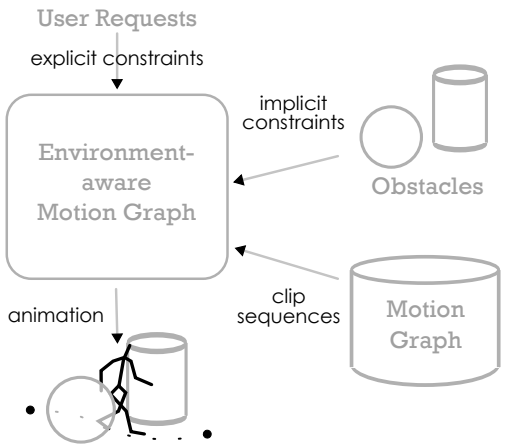


Figure 3.2: The schematic view of the environment-aware motion graph.

constrain each other, causing detoured paths and disrupted animation with excessive transitions. Hence, the decoupling of the motion planning and motion synthesis of the conventional approach is in fact producing animations that may be less plausible.

3.2 Environment-aware Motion Graphs

To prevent detours and frequent transitions in the navigation path, we must consider motion planning and motion synthesis simultaneously. In order to achieve this, we need to be able to integrate motion query with path finding. Only few have attempted investigating this problem. For example, to utilise motion graphs in path finding, Lau and Kuffner (2005) repeatedly expand short clips into the environment to discover a collision-free trajectory to the goal, but they only consider root trajectories so their collision detection is coarse and avoidance is inaccurate. More recently, various reinforcement learning techniques are applied to construct mappings between obstacle configurations and avoidance motions to enable collision avoidance in graph queries (Treuille et al., 2007; Lee et al., 2009; Lo and Zwicker, 2010), but these methods are hard to scale with the number and the complexity of the obstacles. The main concern of these studies is centred on the responsiveness of motion control but none of them discuss the effectiveness and the naturalness of generated animations.

To provide plausible navigation animation, I argue that the precision of collision detection is crucial. My proposal is to incorporate full-body collision detection into motion graph queries. That is, I examine exact movement of possible motions through the graph representation of the known behaviours to determine how close they arrive at the goal (explicit constraints) and how well they avoid the obstacles (implicit constraints). A conceptual view of my combined motion search, called the *environment-aware motion graph*, is illustrated in figure 3.2. The intuition is that by jointly optimising the clip sequence towards being goal-reaching and collision-free, we are able to exploit the most feasible motions to navigate through obstacles without any simplification. Once a solution is found, a smooth-transitioned full-body navigation animation is obtained directly by concatenating the clip sequence without additional approxi-

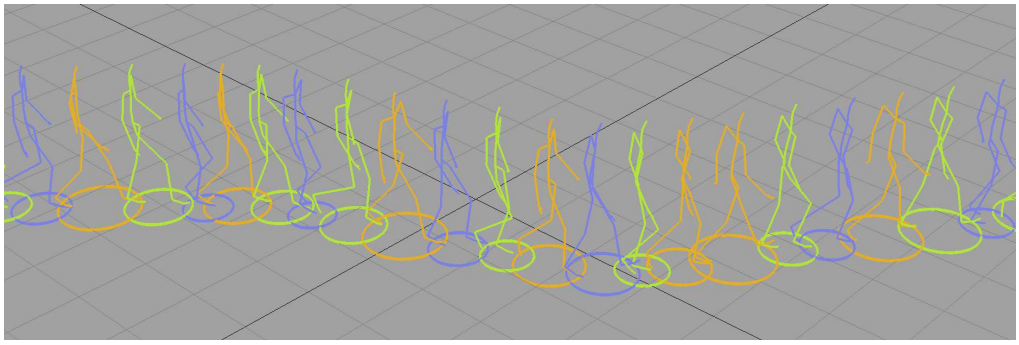


Figure 3.3: The *dynamic bounding circle* associated with each frame. Notice how the radius changes to tightly fit each pose.

mation. That way, we can cast full-body motion planning as a motion graph query problem by integrating collision avoidance into motion concatenation with a joint cost function.

3.2.1 Full-body Collision Detection

To be able to evaluate how well a solution is avoiding obstacles, we need to be able to detect collisions between the solution and obstacles. Given a clip sequence $\bar{x}$ obtained from the graph, we denote the sequence of frames concatenated from the solution as $F = (f_1, f_2, ..., f_n \mid n = \text{number of frames})$ with their corresponding positions $P = (\rho_1, \rho_2, ...\rho_n)$ and yaw-axis rotations $\Theta = (\theta_1, \theta_2, ..., \theta_n)$. Since each frame is defined in local coordinate, we can transform any frame f_i to its global location simply by applying rotation θ_i and then moving to position ρ_i. Since we now have the exact location of each pose, we are able to detect collisions globally and pinpoint individual collisions down to the finest level of limbs if required. I first find the nearest bone to an obstacle to determine whether they are collided. This is done by comparing their minimal distance to the thickness of the bounding capsule of the bone segment. Let $F' = (f'_1, f'_2, ..., f'_n)$ be the transformed global poses of F. The collision error ψ of $\bar{x}$ is calculated by

$$\psi = \sum_{o_i \in O} \sum_{f'_i \in F'} \begin{cases} 1, & \text{if } f'_i \text{ collides with } o_i \\ 0, & \text{otherwise} \end{cases}, \tag{3.3}$$

where O is a set of obstacles $O = \{o_1, o_2, ..., o_m \mid m = \text{number of obstacles}\}$ in the environment. Another benefit is that this method allows us to detect collisions with moving obstacles, which will be addressed in the next chapter.

It is apparent that full-body collision detection is computationally intensive. To minimise the computations, I apply a *broad-phase selection* policy to shortlist a smaller set of frames before precise full-body collision detection for each static obstacle. The broad-phase selection policy uses the precomputed *dynamic bounding circles* associated with each frame to quickly determine whether a pose cab be safely ignored in full-body collision detection by its distance to the obstacle. Unlike maximum bounding circles of the conventional approach (see Section 3.1), dynamic bounding circles are calculated on a per-frame basis. During motion graph initialisation, a minimum bounding circle is computed to tightly fit the pose

of each frame in the local coordinates. Hence, they can be rotated and moved using the same way as transforming a pose to its global location once a clip sequence is concatenated. Figure 3.3 shows an example of a sequence of poses that are associated with their bounding circles. This broad-phase selection policy can effectively reduce the pairs for full-body collision detection to a much smaller number than $m \times n$, particularly in those situations where most obstacles in the environment are relatively small. This is because my collision models are directly constructed using motion data.

3.2.2 Joint Cost Function

Now that we are able to determine collision error for a solution, I begin to describe my joint cost function that combines explicit and implicit constraints. Given a goal location q, the cost of a solution $\bar{x}$ is calculated by

$$C_{\text{joint}}(\bar{x}) = \overbrace{\underbrace{\frac{|p-q|}{\Delta}}_{\text{deviation}}}^{\text{explicit}} + \overbrace{\underbrace{\psi}_{\text{collision}} + \underbrace{max(\frac{\lambda}{\Lambda}, 1)}_{\text{distance}} + \underbrace{max(\frac{\tau}{\mathrm{T}}, 1)}_{\text{duration}}}^{\text{implicit}}, \tag{3.4}$$

where p, ψ, λ, and τ are the final location, the number of collisions, the travelled distance, and the duration of $\bar{x}$, respectively. Δ is a scale parameter to modulate the deviation, and Λ and T are the least distance and time required to travel along the straight line between the initial and goal locations. The first part of the equation is the explicit constraint that penalises the solution by measuring the deviation between final location p and goal location q. We can also integrate intermediate explicit constraints, such as the waypoint deviation of equation 3.1, if we are required to travel through multiple waypoints. The positive non-zero scalar Δ is often chosen with a short distance such as 20 centimetres to soften the deviation so that we do not discriminate implicit costs but still favour a solution that arrives close to the destination.

The second part of the cost function is three implicit constraints that are essential for efficient navigation and collision avoidance. Previously, I have described how to obtain collision error ψ for a solution $\bar{x}$ in Equation 3.3. By integrating ψ in the cost function, I penalise the solution to avoid collisions. However, in order to avoid obstacles, it is reasonable to assume that extra transitions such as turning or slowing-down motions are needed. From time to time the search algorithm might inadvertently choose a solution that requires more then necessary motions to reduce collision errors. To minimise detours and delays, I penalise the solution by calculating the ratio of the actual travel distance (time) to that of a physically possible minimum distance (time). The actual travel distance λ is obtained by measuring the arclength of the ground trajectory of solution $\bar{x}$. λ is then divided by an estimated minimum distance Λ, which is the length of the straight line between initial and goal locations, as the travel distance ratio. This ratio is then clamped to 1 to avoid choosing a solution that is far too short. Similarly, the ratio for duration cost is found by using an estimated minimum time $\mathrm{T} = \frac{\Lambda}{\mathrm{V}_{\text{max}}}$ where V_{max} is the maximum moving speed found in the motion database. The aim of the search algorithm is then to find a solution of which both the ratios are close to one. Additionally, we could cancel the minimal value 1 for the distance ratio such that we optimise a path by minimising its summarised cost close to zero. Therefore,

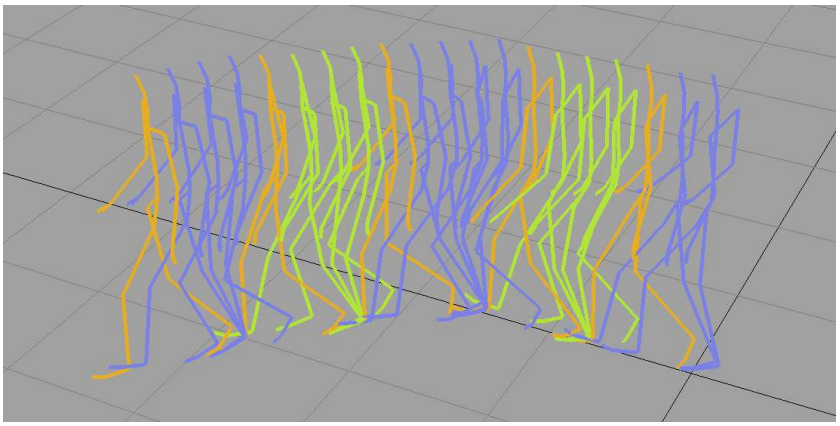


Figure 3.4: Each pose is labelled with a corresponding supporting-foot phase such as left-foot (green), right-foot (indigo), and double-support (orange).

we are able to search for an efficient and collision-free full-body navigation animation using a joint cost function.

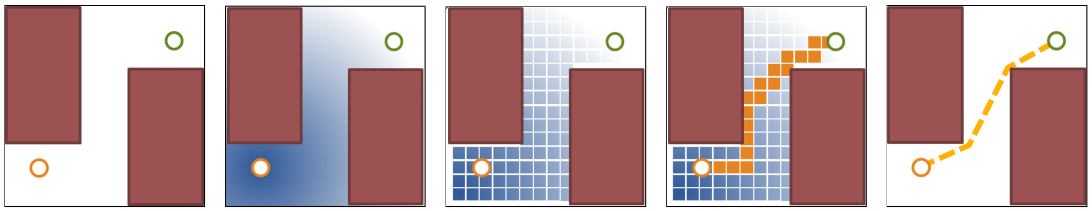
3.3 Implementation

To realise the conventional decoupled approach and my combined method, I implemented the **randomised motion graph search** algorithm that accepts both cost functions with their constraints (see Section 3.3.1). For the path-finding stage of the decoupled approach, I also implemented the **potential-field path finding** technique to retrieve a collision-free ground trajectory that to be used to query motions in the path-following stage (Section 3.3.2). These algorithms are implemented as Python interpreter modules with core procedures writing using the C++ programming language. The generated animations are visualised in the Maya animation software.

3.3.1 Randomised Motion Graph Search

As described in Section 3.1, to query the motion graph, we require a cost function and a search algorithm to evaluate a solution traversed from the graph and to find a solution that minimises the cost, respectively. In previous sections, I have discussed both cost functions for the decoupled and combined search. Here, I describe the search algorithm implemented in this thesis. This search algorithm is based on the randomised motion graph search of Arikan and Forsyth (2002). I extend their graph search to support path-following and my joint cost function to enable navigation and collision avoidance. Thus, I only consider location constraints in my cost functions, although other types constraints such as rotation and timing can also be taken into account, as described in the standard motion graph search.

I first assume that we have a motion graph built by connecting transitions between similar clips in a collection of motions. I further assume that these motions are labelled with different phases, such as left-foot, right-foot, and double support, to eliminate the transitions that are out-of-phase. During graph construction, I apply a supervised learning technique (Ikemoto et al., 2006) to label each frame with



(a) (b) (c) (d) (e)

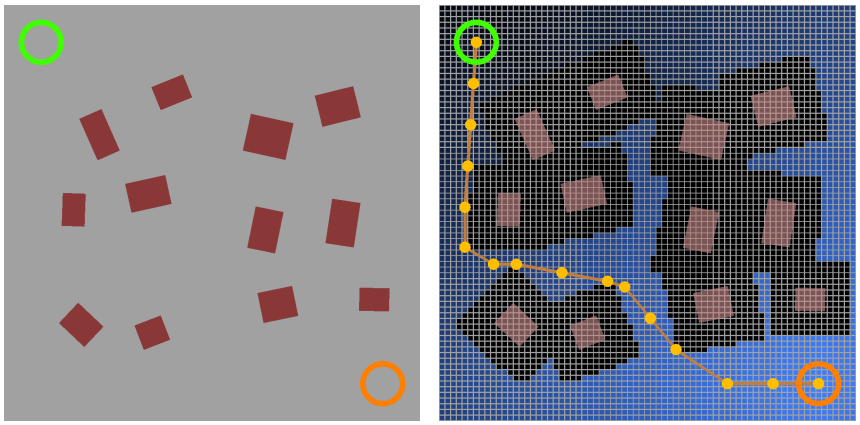
Figure 3.5: Potential-field path finding: (a) Initial location (green) and goal location (orange) among obstacles (red). (b) The gradient circle of the potential field centred at the goal. (c) Potentials are stored in discretised free-space cells. (d) Potentials are used as heuristics to search for a chain of cells (in orange) connecting to the initial location. (e) A trajectory is converted and straightened from the chain.

corresponding supporting-foot phases (see figure 3.4). In my motion graph implementation, I only allow transitions in single-support phases such that I am able to enforce planting on the supporting foot when blending transitions in the final animation. Another benefit is that we can prune down the size of the graph which is useful to reduce the complexity of graph search.

The randomised motion graph search algorithm is a simulation-based optimisation. To search the graph, the algorithm requires a pair of starting and end poses (hard constraints) to find a series of transitions, also referred as a "path", that connects these poses. For example, we can specify a frame at the beginning of a "stand-to-walk" motion and another frame at the end of a "walk-to-stand" motion as the starting and end poses to traverse across the graph for other walking motion clips between them. Since a database such as a walking motion graph contains many similar clips, normally there exist infinite alternative paths between these two poses. The graph search begins by randomly choosing one (a seed) and then deliberately replacing its sub-paths to explore other alternative transitions (mutations). The path is optimised by a series of Monte Carlo simulations which iteratively enumerate mutations and score them using the cost function (soft constraints) until no alternative transitions can be found to minimise the cost. Whilst it does not guarantee the global optimality of the search due to the nature of motion graphs being cyclic, this algorithm provides an effective way for the convergence of multiple seed samples to many local optima that fulfil the same set of constraints. Thus, we can increase the number of samples and choose the local optimum solution that minimises the cost.

3.3.2 Potential-field Path Finding

For path-finding, I chose to implement the potential fields method (Barraquand et al., 1991) as it is considered to be effective in low-dimensional path-finding problems, such as 2-D trajectory planning. As illustrated in figure 3.5, the potential-field method finds a path by propagating a circular gradient of ascending distances, the potentials, in the free space from the goal location in the discretised workspace of a grid of cells. The resolution of the grid, which determines the quality of the path, is defined by a cell size D_{cell}. While a larger D_{cell} can reduce the search space, a smaller D_{cell} creates a finer approximation to the environment. The cells in this field, which are filled with potential values, are then used as



(a) Obstacles (red boxes). (b) The trajectory (yellow lines).

Figure 3.6: An example of planning a trajectory using potential fields. The black cells are determined using a obstacle-growing technique (Lozano-Pérez and Wesley, 1979) to keep the trajectory away from obstacles.

heuristics to search for a connected chain of cells to the initial location by incrementally moving to a lower potential neighbour cell. The chain is converted into a trajectory by connecting the centroids of consecutive cells. The trajectory is further straightened, known as "path smoothing", if any two consecutive segments can be joined as a straight line without intersecting with any obstacles. By iterating through the segments of the path and smoothing them, we can reduce the length of the trajectory close to the possible shortest path.

As described in Section 3.1, we cannot directly query an animation to follow this trajectory since it does not consider the volume of the virtual human. To workaround this, I apply the obstacle-growing technique (Lozano-Pérez and Wesley, 1979) to expand obstacles outwards with a distance of the maximum bounding radius R_{max} estimated from the maximal minimum bounding circles in the motion library before computing the potential field. In my implementation, I also include a distance D_{off} to allow a small amount of deviation when querying motions to follow the trajectory. That way we guarantee a trajectory always stays at least $(R_{\text{max}} + D_{\text{off}})$ away from all obstacles. Figure 3.6 demonstrates an example of a trajectory planned with the expanded obstacles. At path-following stage, waypoints are planted along the trajectory for every step size D_{step} to confine the animation to stay within the corridor formed by the trajectory with the width of $(2 \times D_{\text{off}})$. Hence, we are able to query the motion graph using equation 3.1 with these waypoints and the goal by assigning $\Delta = D_{\text{off}}$.

3.4 Experimental Results

To compare my combined method with conventional decoupled approach, I have generated test navigation animations in various experiments involving different obstacle situations. Here, I use symbolic names PFF and EMG to refer the conventional decoupled approach, path finding and following, and my combined search, environment-aware motion graphs, respectively. The hypothesis of the experiment is

stated as follows:

> Incorporating full-body collision detection in motion graph queries will increase the accuracy of avoidance in the generation of navigation animation, which improves the plausibility of the resulting animations in that it minimises the *travel distance and travel time* as well as the *transition rate and transition cost* to avoid obstacles and reach the goal.

The PFF approach generates navigation animations by approximating motions on a trajectory that is planned using a simplified collision model. Hence the collision detection and avoidance are less accurate. The EMG method integrates collision avoidance into motion graph queries such that it supports full-body collision detection which is relatively more accurate. The results generated by both methods are compared by the following measurements:

- **Travel distance:** the arclength of the ground trajectory of the root (pelvis) joint.
- **Travel time:** the duration of the animation.
- **Transition rate:** the frequency of switching between motion clips.
- **Transition cost:** the total "dissimilarities" of the transitions between motion clips.

The travel distance and travel time are to measure the required journey of the animation while the transition rate and cost are to determine the smoothness of the solution. The transition rate is defined by dividing the number of transitions by the travel time to indicate how many transition is made per second. A higher transition rate implies frequent pace or course changes, which are often considered as a potential anomalous behaviour. The transition cost measurement is to indicate the smoothness of a solution. During motion graph construction, each transition is associated with a cost that measures the dissimilarity between two poses of the transition (see Section 2.2). Thus, a lower cost means that two motions connected by the transition are relatively similar and hence the animation is more likely to be smooth after blending.

To generate locomotion animations, I built a motion graph of human walking motions to query animation for the path-following stage in PFF and for EMG. The motion data used to construct the motion graph from are selected from the CMU motion capture database (CMU Graphics Lab, 2009) and are listed as Table 3.1. All of them are re-sampled to 30 frames per second. Each of them represents a specific walking action, such as turning in different angles and change in pace. Therefore, a transition between two clips can be seen as the indication of the change of behaviours.

For the settings of the potential-field in PFF, I tried different D_{cell} between $1 - 20cm$ and found $5cm$ to be a reasonable size that can produce a fine approximation of the environment in my experiments without significantly increasing search time. I also found maximum bounding radius $R_{\text{max}} = 42.7cm$ in the walking motion database to be the distance to expand obstacles. To confine the animation stay in the corridor formed by the trajectory, I chose $D_{\text{step}} = 20cm$ to generate waypoints to be planted along the trajectory. The deviation scalar Δ is also set to be $20cm$ for each waypoint and the goal (see Equation 3.1). I then compensate the allowable deviation $D_{\text{off}} = 20cm$ to the obstacle expansion such that the

CMU trial no.	Description
16_12	walk, veer left
16_14	walk, veer right
16_18	walk, 90-degree left turn
16_19	walk, 90-degree right turn
16_31	stop to walk
16_34	walk to stop
16_47	walk in a straight line
16_58	walk in a straight line

Table 3.1: The motion capture data used to construct the "walking" motion graph.

trajectory is at least $62.7cm$ away from any obstacle. Note that in order to stay close to the path, unlike standard motion graph search, I did not take into account transition cost in the cost function. Although this might sacrifice smoothness the results did not show any noticeable artifacts after.

The EMG method does not require a path-planning stage in the experiments, although certain waypoints may be required to prevent deadlocks when navigating in larger and more complex scenes such as maze-like terrains. For the experiments in this chapter, I also chose the deviation scalar $\Delta = 20cm$ in equation 3.4 to match the settings in PFF.

I have designed a **two-corner turning** (Section 3.4.1) test scene and a series of **object-cluttered space** (Section 3.4.2) test scenes to evaluate the navigation capability at different level of scene complexity. Both PFF and EMG are able to find solutions that arrive close to the goal without collisions in the experiments using the aforementioned settings. All experiments were performed on an Intel P9700 2.8 GHz processor on a 64-bit Windows laptop machine with 6 GB memory. The videos of the results of these experiments are available in the supplementary video, as detailed in Appendix A.1.

3.4.1 Two-corner Turning

In this experiment, I designed a test scene that comprised two large obstacles which form a corridor with two consecutive turns, as seen in figure 3.7. The initial and goal locations are situated at both ends of the corridor. Both PFF (indigo) and EMG (orange) are able to find a solution turns around the corners and arrives at the goal. Figure 3.9(a) shows the root trajectories generated by both methods. The animation of PFF is queried by approximating the path generated by a potential field as shown in Figure 3.8. Both PFF and EMG can arrive very closely to the goal under the specified deviation scalar Δ ($20cm$). As PFF strictly follows the path by making sharp turns, EMG can generate a relatively smooth turning motion at the first corner, which also reduces travel distance. It is also noticeable that in order to closely approximate the path, PFF tends to select motions with slower speed and requires extra transitions to constantly adjust the steps and the orientation before the corner such that it can stay within the allowable deviation regions. As a result, the distance between PFF and EMG increases after each corner as can be seen in Figure 3.7.

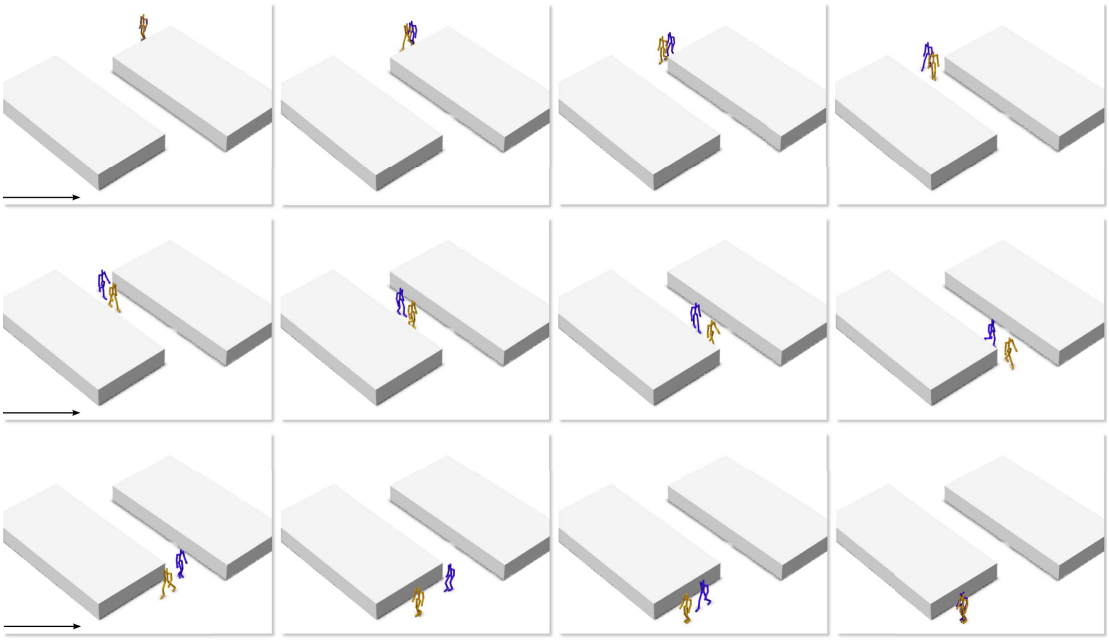


Figure 3.7: A snapshot sequence of two-corner turning. The distance between PFF (indigo) and EMG (orange) gradually increases after each corner.

Figure 3.9(b) shows the quantitative comparison between the results generated by both methods. The results show that EMG performs better in four measurements. EMG requires at least 1 metre shorter and 1 second sooner than PFF. The transition rate and the transition cost of EMG were both lower than PFF, so the overall smoothness of EMG is also relatively better.

3.4.2 Object-cluttered Space

In the second experiment, I evaluated both methods by navigating through multiple smaller box obstacles. I first arranged a grid of 4×3 boxes ($45 \times 90 \times 45cm$) that are uniformly distributed in a $720 \times 540cm$ rectangular region (see figure 3.10). The interval between each pair of adjacent boxes was $180cm$ with an actual clearance of $180 - 45 = 135cm$, which was slightly larger than the expanded offset from both obstacles $2 \times (42.7 + 20) = 125.4cm$ required by path-finding in PFF. Figure 3.11 shows the path found using the potential field method while Figure 3.12(a) shows the root trajectories of the animations generated by both methods. Both PFF and EMG chose the similar routes along the diagonal. While PFF followed the zigzag-shape path to navigate though obstacles, EMG can closely pass obstacles by walking along a much straighter line to the goal. Figure 3.12(b) compares the quantitative measurements of both results. PFF suffered from strictly traversing the erratic path planned by the potential field method such that a longer distance and time to complete the journey. While the transition rate of EMG was slightly higher, the transition cost of EMG was still lower than PFF.

Next, I rearranged the obstacles by randomly moving around their locations and changing their size and orientation, as shown in Figure 3.13. I repeat this randomisation using different configurations

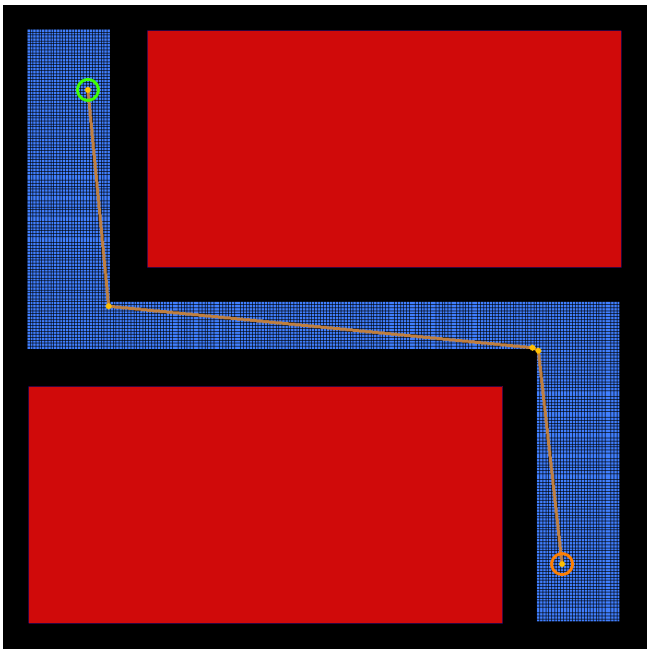


Figure 3.8: The potential-field path-finding in two-corner turning. The obstacles (red) are expanded (black regions) to keep the path away.

for nine trials and report their root trajectories and quantitative comparisons of the results generated by both methods from Figure 3.15 to Figure 3.23. Overall, EMG can find shorter paths to the goal while PFF can only travel along paths through larger gaps between obstacles. Figure 3.14 shows an example where PFF was only able to find a path through a few large gaps on the right hand side of the test scene. While EMG can travel along paths that are more close to the diagonal line, PFF was not able to find a similar short route. In fact, only three out of nine random configurations, PFF was able to find a path that travels through the obstacles cluttered in the middle of the test scenes (see Figure 3.16, 3.21, 3.23). From quantitative comparisons we can see that EMG dominated all measurements in nine random trials, with only one minor exception which is the transition rate of random config. 7 (Figure 3.21).

3.5 Summary

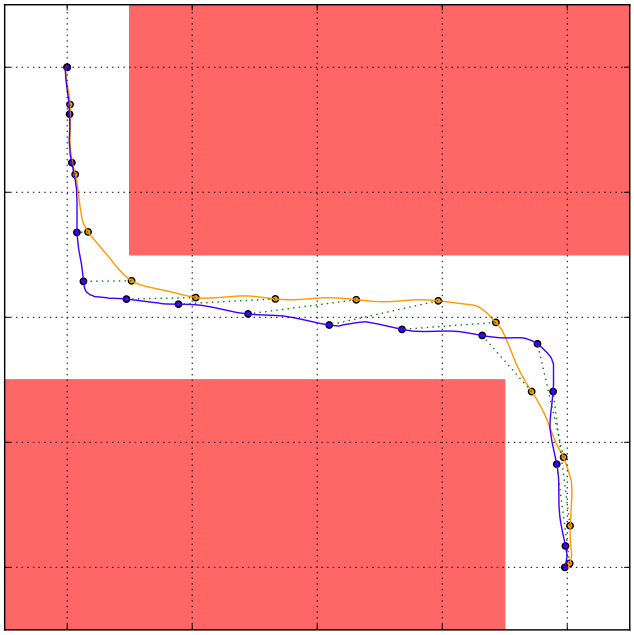
In this chapter, I have presented a joint optimisation method that enables motion planning in motion database query for character navigation and collision avoidance. This technique supports precise collision detection and smooth motion concatenation which allow us to generate plausible and collision-free full-body animations that travel through obstacles and reach the goal. The results generated using my combined search are more plausible than the conventional decoupled approach in that it minimises the anomalous behaviours such as excessive travel paths and behaviour changes. The main reason for this is that my joint optimisation considers motion planning and motion synthesis simultaneously. Unlike the conventional decoupled approach, my method does not require to fit motions on a trajectory that is conservatively planned using simplified collision models. Therefore, my approach avoids a step that would

otherwise produces an animation that follows an erratic path with frequent pace and course changes.

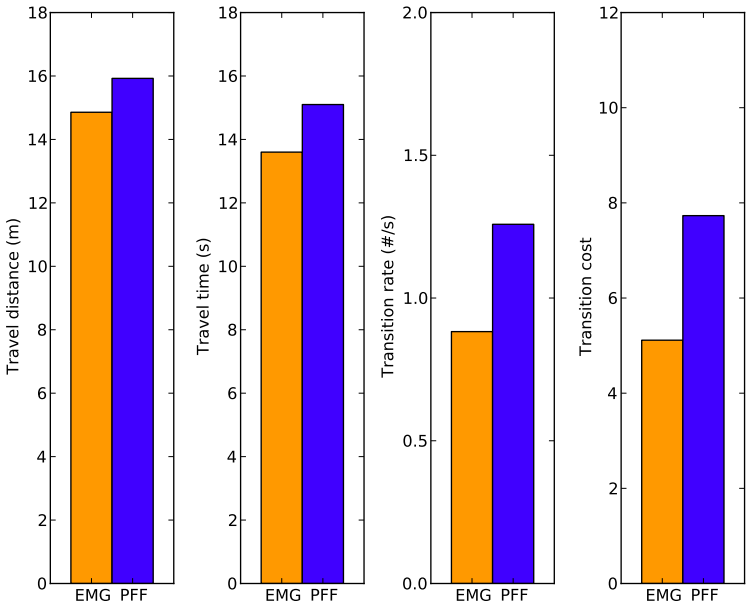
My motion query technique supports path planning by incorporating collision errors in a joint cost function through a randomised motion graph search. This combination allows us to optimise a navigation animation by choosing the most feasible motion sequences that avoid obstacles using full-body collision detection. Once a collision-free solution is found, it can be directly concatenated into a smoothly transitioned full-body animation. I view collision avoidance a soft-constrained optimisation. Therefore, the joint cost function is able to take into account multiple obstacles as soft constraints. This also allows for the collision avoidance of moving obstacles, which will be described in the next chapter. In order to avoid obstacles more precisely, this technique relies on full-body collision detection, which is a computationally demanding task. I address this by pre-computing a dynamic bounding circle for each pose in the motion database to allow a broad-phase obstacle selection. This reduces the number of full-body collision detection required.

Whilst my technique can reduce anomalous behaviours in navigation animations, this method also has several limitations. My motion graph queries avoids collisions by modelling obstacles as soft constraints and minimising the collision errors in the joint cost function. One fundamental limitation for this method is that it cannot guarantee a collision-free solution, although the results show that my technique was able to find animations that avoid obstalces in all the experiments. Another concern is the time complexity of the motion graph search. Generally speaking, the search complexity of a motion graph increases exponentially with the size of the graph as well as the length of the animation. Thus it is always an important issue to balance between the size and the motion variety of the motion graph. Since the optimality of the randomised motion graph query relies on the number of samples (random seeds), the search time can be problematic when we are required to generate long navigation animations that navigate through large scenes. One potential solution to this is to decompose the query into several subqueries such that we can plan long navigation animations incrementally. The number of waypoints can also be an issue when planning long navigation animations. This is because every pose in the solution has to be examined in order to find the closest one to a waypoint, and this repeats for each waypoint until the final goal. One possible solution to this would be to consider some spatial partitioning technique to index the position of the poses more efficiently.

The aim of this study is to generate navigation animations that behave more plausibly. I define an animation to be more plausible if it requires a shorter path and a smoother sequence of motions to reach the same goal. The smoothness of the animation is partly judged by the transition rate of the motion sequence to determine how frequently a transition is made, i.e. switching between motion clips. However, this might not always be a good indicator as switching to another motion does not necessarily mean it is changing to a different behaviour. In Chapter 5 I will discuss a new approach that evaluates the smoothness of the animation based on the kinematic properties of the curvilinear motions.



(a) Root trajectories of PFF (indigo) and EMG (orange), with green dashed lines representing the corresponding root location recorded at the same time step.



(b) Travel and transition comparisons.

Figure 3.9: The results of the two-corner turning test scene: (a) the distance between PFF and EMG (indicated by the green dashed lines) gradually increased. (b) EMG requires less travel distance and travel time and lower transition rate and transition cost than PFF.

Figure 3.10: A snapshot sequence of a grid of boxes. PFF (indigo) traverses a zig-zag path (see Figure 3.11 while EMG stays close to the shorter diagonal line.)

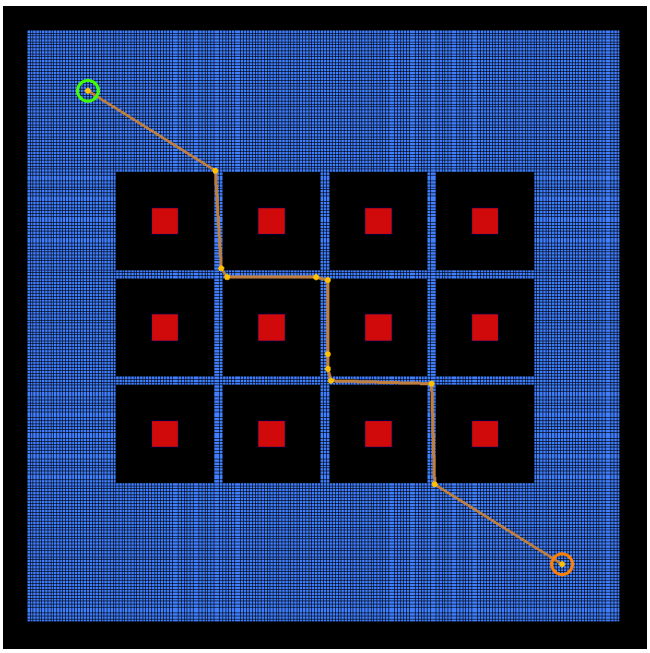
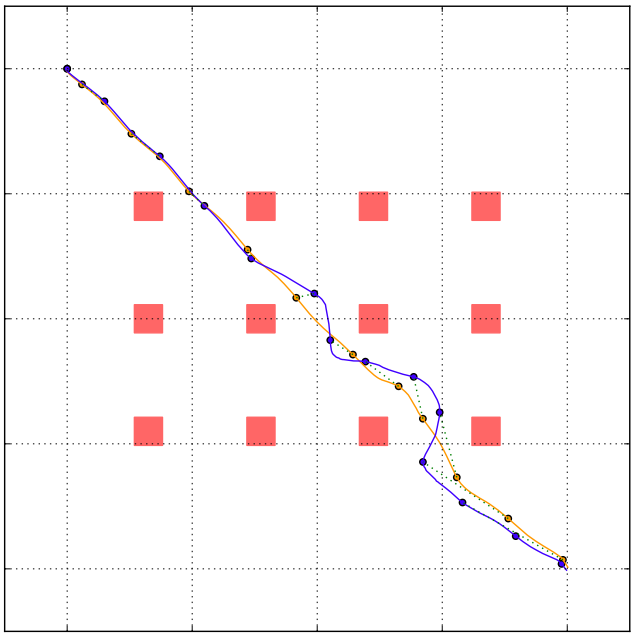
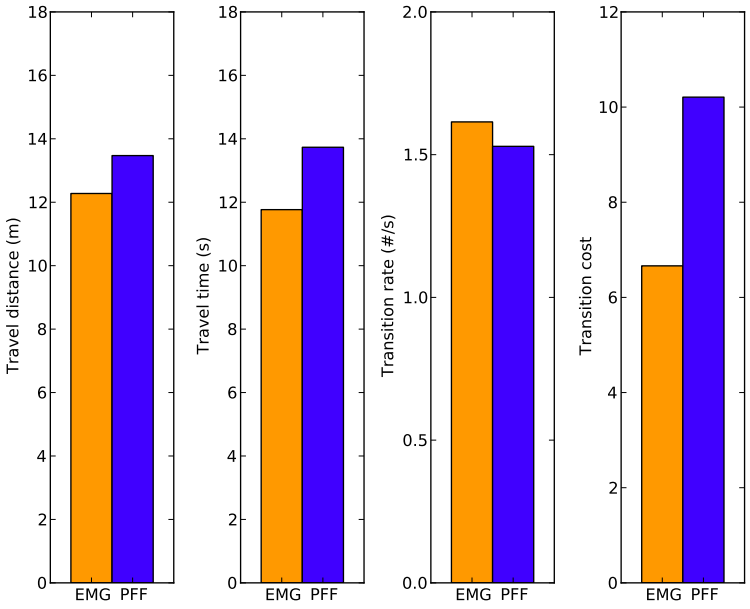


Figure 3.11: The potential-field path finding in a grid of boxes. The expansion of the obstacles forces the path finder to choose an erratic zig-zag path.



(a) Root trajectories. Notice that at PFF can only remain straight while stay with the allowable deviation zone for the first half of the journey. EMG used a much straighter line to reach the goal.



(b) Travel and transition comparisons. Both the travel distance and travel time of EMG were lower than PFF. Although EMG required slightly higher transition rate than PFF, the transition cost of EMG is still lower.

Figure 3.12: The results of the grid-of-box test scene in object-cluttered space.

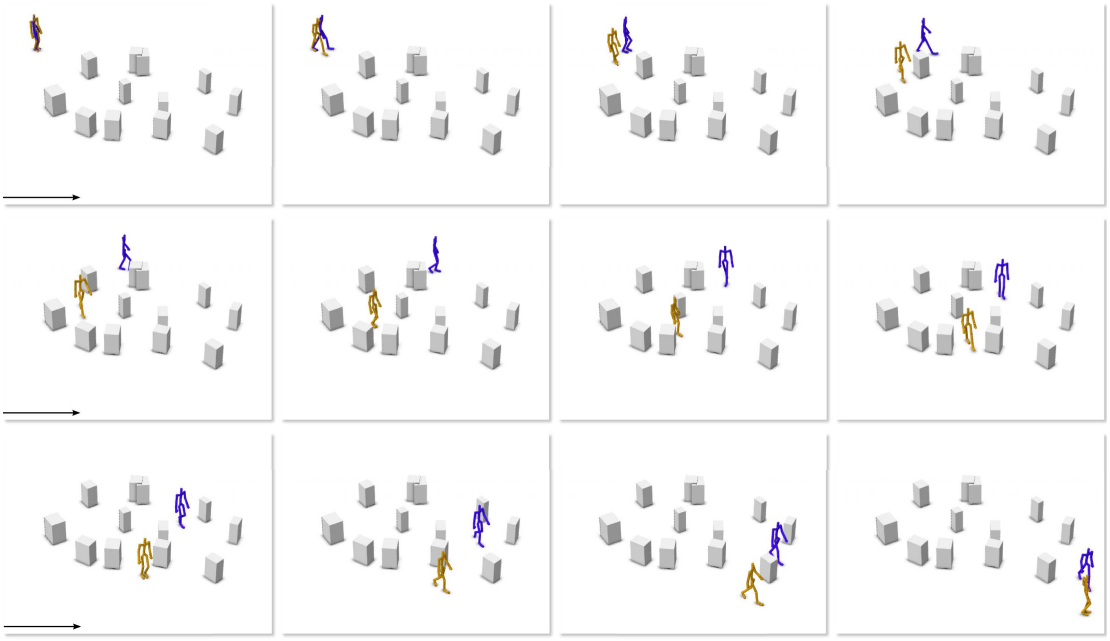


Figure 3.13: A snapshot sequence of an example object-cluttered space test scene (Figure 3.21). Whilst both PFF and EMG can travel around obstacles in this experiment, EMG is able to find a shorter path through smaller clearances between obstacles.

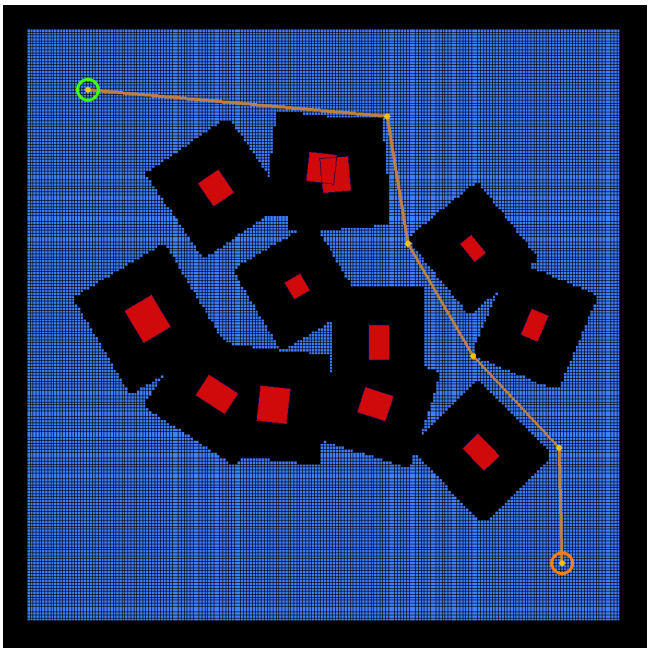
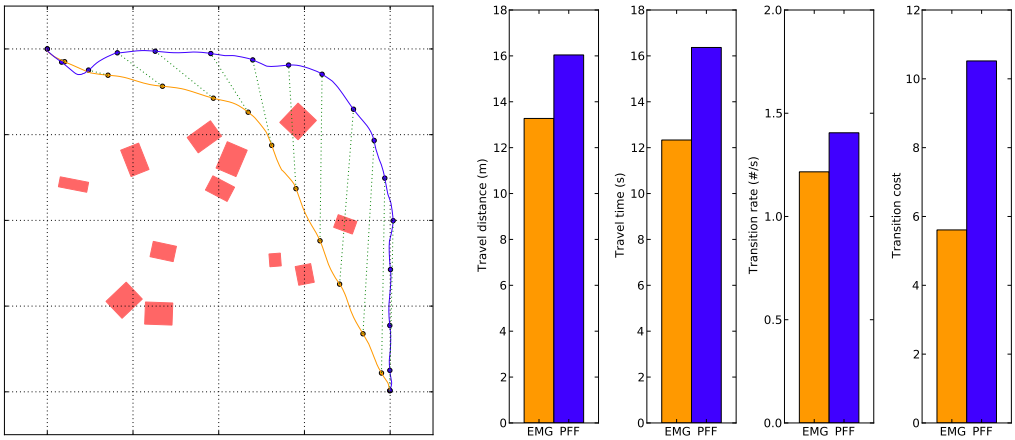
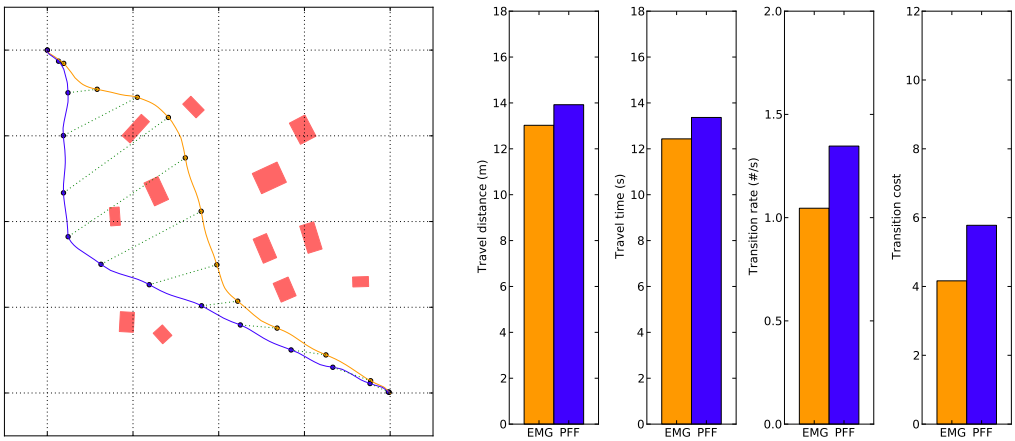


Figure 3.14: The potential-field path finding in an example randomised object-cluttered space test scene (see Figure 3.21 for the comparisons of the results).



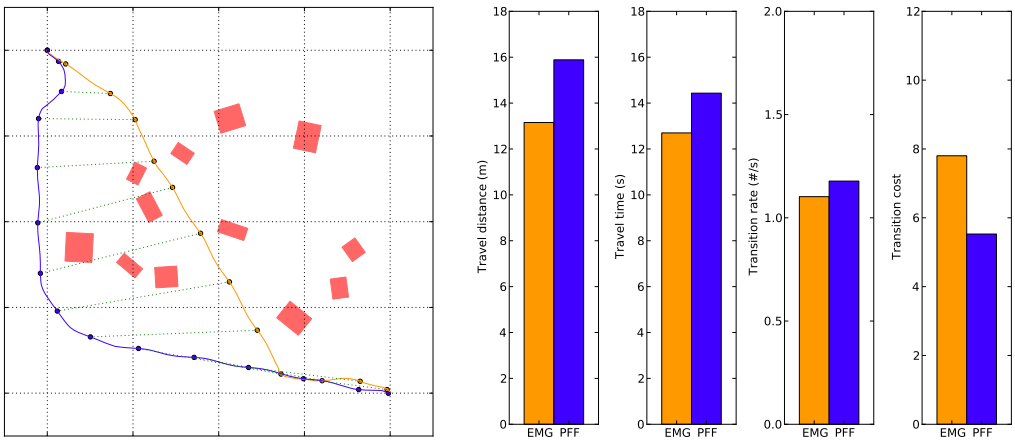
(a) Root trajectories (b) Travel and transition comparisons

Figure 3.15: The results of random config. 1.



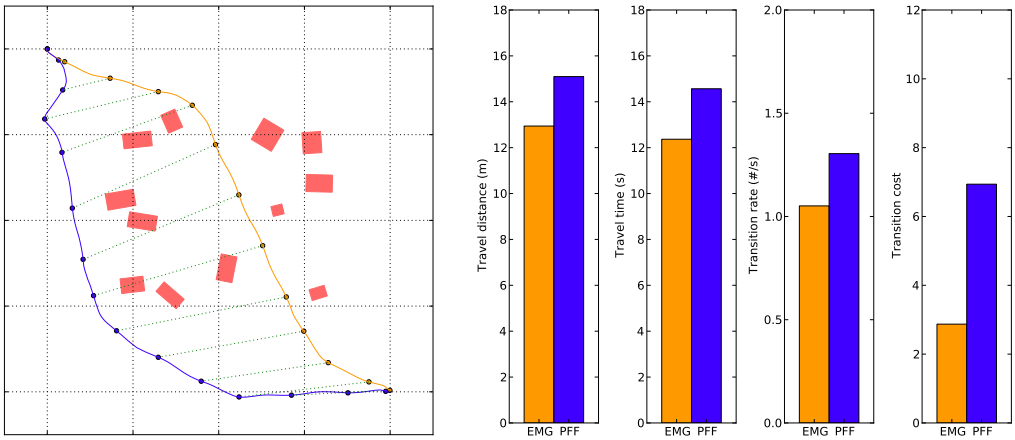
(a) Root trajectories (b) Travel and transition comparisons

Figure 3.16: The results of random config. 2.



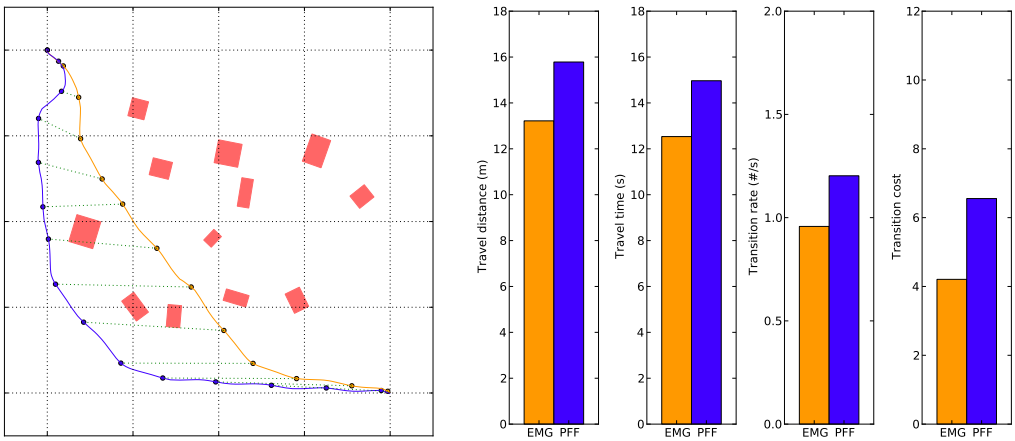
(a) Root trajectories (b) Travel and transition comparisons

Figure 3.17: The results of random config. 3.



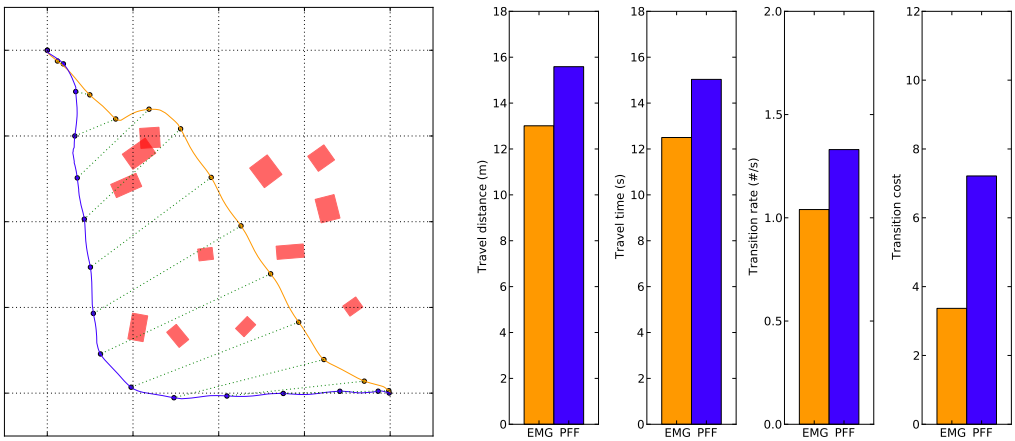
(a) Root trajectories (b) Travel and transition comparisons

Figure 3.18: The results of random config. 4.



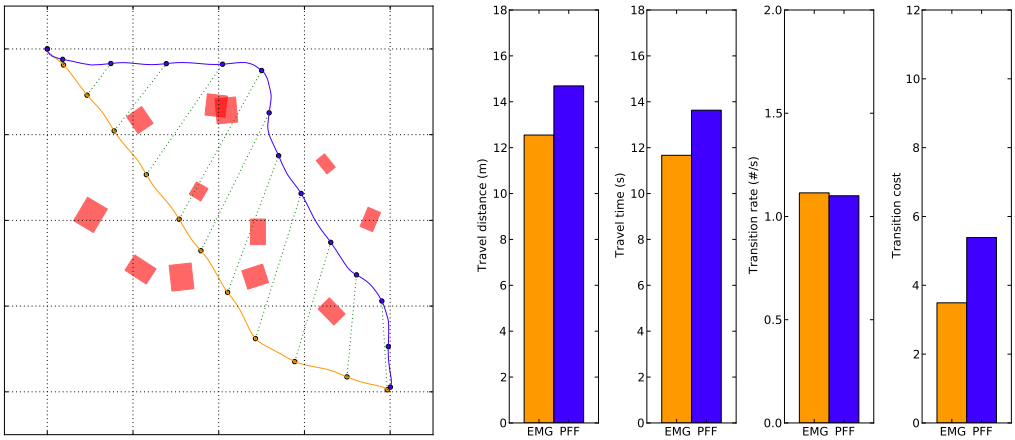
(a) Root trajectories (b) Travel and transition comparisons

Figure 3.19: The results of random config. 5.



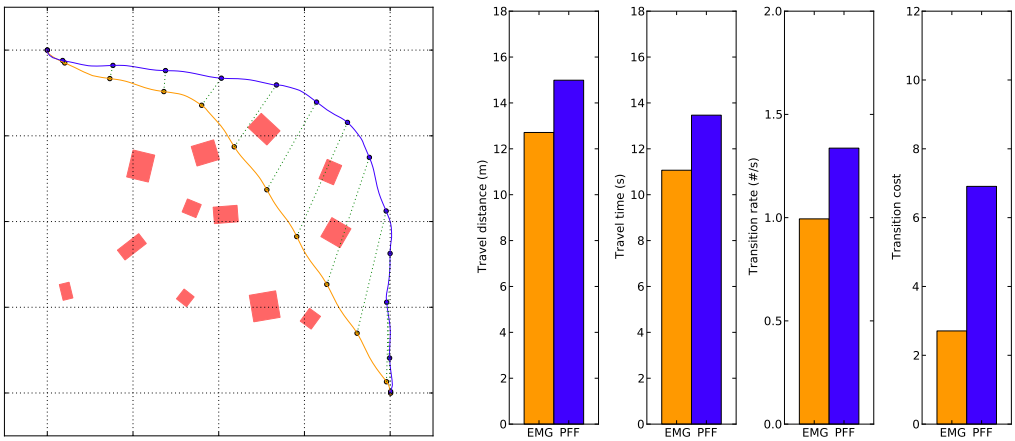
(a) Root trajectories (b) Travel and transition comparisons

Figure 3.20: The results of random config. 6



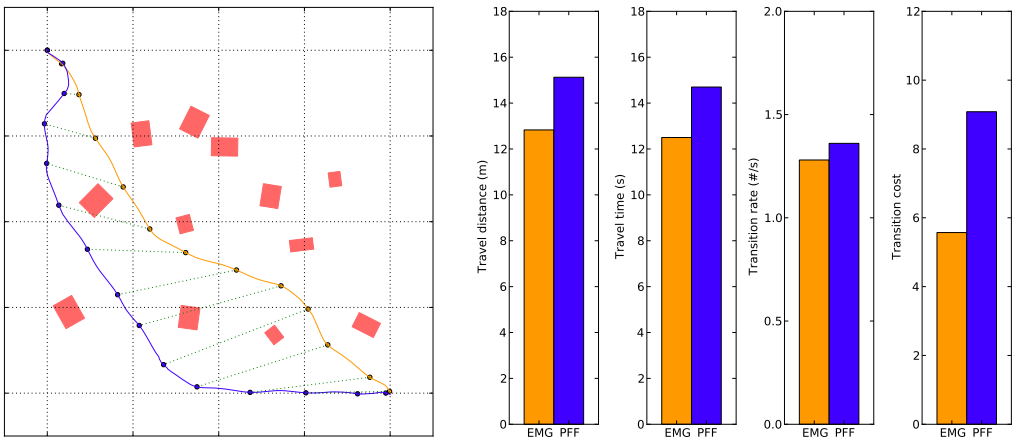
(a) Root trajectories (b) Travel and transition comparisons

Figure 3.21: The results of random config. 7



(a) Root trajectories (b) Travel and transition comparisons

Figure 3.22: The results of random config. 8



(a) Root trajectories (b) Travel and transition comparisons

Figure 3.23: The results of random config. 9

Chapter 4

Multi-character Navigation

The creation of navigation animations for multiple characters is often solved by planning individual motion paths and then querying animations to traverse these paths. Whilst fitting motions onto paths itself is problematic, as discussed in Chapter 3, to collectively plan interactions between individuals when many of their paths interfere is very difficult. One challenge is to find each individual a path heading towards the goal while avoiding others en route. To find such paths in crowded situations is a hard problem because the movement of each individual is interdependent.

Previously, interpersonal collision avoidance was considered independently from the path planning of global navigation towards the goals. Unlike goal navigation, which is usually planned beforehand, the collision avoidance between individuals is calculated as a secondary reinforcement to separate proximate neighbours apart on a per-frame basis. Such collision avoidance approach is not optimal in that it only relies on the instantaneous movements of the individuals in near proximity. This local collision-avoidance strategy lacks the anticipation of collisions which often leads to congestions and deadlock situations. These consequences in turn causes sub-optimal animations such as choosing an erratic path and making excessive behaviour changes, despite the optimality of the global navigation planned in the first place.

In this research, I argue that to find plausible animations for crowd navigation, close-proximity interactions between individuals are crucial. Therefore, my proposal is to solve the planning of global navigation and the interpersonal collision avoidance simultaneously. To unify both, the planning of crowd navigation is formulated as an iterative approximation that incrementally localises interfered paths of individuals and plans their close interactions to avoid each other. By identifying the spatio-temporal dependencies between collisions and avoiding them with precise collision detection, this technique is able to minimise the disturbances caused by resolving the interferences between individual paths. The results demonstrate that my incremental planning framework can effectively reduce anomalous collision avoidance behaviours in navigation animations that are problematic in the other approaches such as frequent course or pace changes. Furthermore, since each path is planned through direct motion database queries, individual animation is visually plausible in that it does not contain obvious artefacts such as foot skating.

In this chapter, I will first discuss conventional crowd simulations and their limitations on planning global navigation. I will then describe how I unify the planning of goal navigation and interpersonal collision avoidance by addressing this as an incremental optimisation problem. My incremental crowd navigation planning technique will be evaluated by comparing the results generated by conventional collision-avoidance models in several classical crowd situations. The applications and limitations of my technique is discussed in the end of this chapter.

4.1 Interdependency of Interpersonal Interactions

The creation of crowd motions is often formulated by simulating the emergent behaviours of the interaction between individuals. For instance, intra-group bonds are joined amongst the individuals to form group formations such as flocks and herds (Reynolds, 1987). Individual identities can also be assigned across the group to enable leadership or adversarial interactions such as pursuit and evasion (Reynolds, 1999). This is commonly realised as a simulation of a dynamical system of crowd individuals from one state to another over the course of time. At each timestep, the state of the motion of each individual is changed according to the movements of the crowd from the previous state. Each of the individuals then determines how to react to others under certain pre-defined local navigation policies. Since the decision of individual motion is interdependent, the temporal interval, or the timestep, between two consecutive states is chosen to be a short period of time so that every member can respond to the change of the proximities more accurately. Therefore, the evolution of the crowd motions depends on the resolution of interactions solely upon the *instantaneous movements* of the previous state. Such property makes the planning of crowd interactions very difficult.

One crucial behaviour that is needed to simulate plausible crowd motions is interpersonal collision avoidance. This is often solved by using local navigation techniques that examine the movements of proximate neighbours and determine how to move away from them. Such local collision-avoidance polices can be as simple as applying repulsive forces computed by mutual distances and bearings to separate neighbours (Reynolds, 1987; Helbing and Molnar, 1995; Reynolds, 1999), but this implementation can cause anomalous behaviours where individuals constantly repel each other. More complex collision-avoidance models such as velocity obstacles (van den Berg et al., 2008a,b; Guy et al., 2009), reinforcement learning-based control policies (Treuille et al., 2007), experiment-based models (Pettré et al., 2009), or synthetic visual perception-based rules (Ondřej et al., 2010) are investigated to enhance the making of the decision to void each other more effectively and more reasonably. However, like other types of interaction, these methods all rely on the instantaneous information of a single state to avoid collisions. Furthermore, crowds in these models are often treated as particle-like entities and thus their collision models are very coarse (such as discs). As a result, to minimise collisions their avoidance strategies must be conservative, i.e., to stay away from others or to slow down as much as possible. In crowded situations these local navigation models can show instability and cause congestions or even deadlocks due to the lack of anticipation to the collision and the inaccuracy of the collision models. The

results for this are anomalous behaviours where individuals erratically change their courses and paces in order to react to the changes of neighbours' movements.

To avoid congestions and deadlock situations, one must be able to control the flow of the crowd. This requires to be able to plan each individual a global path that minimises the obstructions to the destination. Ideally, we could apply path-finding techniques to plan one feasible route for each individual by considering every obstacle including others in the crowd. Unfortunately, it is difficult to do so as once the paths of individuals interfere, their movements become interdependent. One common way of tackling this is to pre-compute global paths as certain guidances, such as navigation graphs (Pettré et al., 2005; Morini et al., 2007) and roadmaps (Kamphuis and Overmars, 2004; Nieuwenhuisen et al., 2007), to steer individuals in the dynamical simulation combined with local navigation policies. To ensure no collisions occur, these algorithms have to prioritise collision avoidance over goal navigation. As a result, the dominance of local navigation often compromises global paths and leads to regional traffic bottlenecks. More sophisticated models, such as dynamic potential fields (Treuille et al., 2006), are applied to smooth the contradictory steering forces between global and local navigation and to disperse the local density of the crowded regions. But they cannot eliminate collisions and thus a secondary local navigation is still required to separate proximate neighbours apart (Narain et al., 2009). Overall, present crowd navigation techniques cannot to anticipate and resolve collisions correctly and as a result they can cause anomalous collision-avoidance behaviours in navigation animations.

4.2 Path Interference Perturbations

To prevent anomalous behaviours in crowd navigation, I propose to incorporate interpersonal collision avoidance into the planning of goal navigation. I formulate this problem as a series of modifications, called *perturbations*, to a set of individual complete animations representing the global paths of the crowd members, in hopes of resolving their *interferences* where individuals collide. To achieve this, I first reduce the problem by finding each individual an optimal path regardless other members in the crowd, referred as the *initial approximation*. The aim of the optimisation is then to reduce the interferences between paths in the approximated solution while minimising the perturbations to resolve the collisions. Throughout the optimisation, I incrementally localise interfered paths as independent subsets and perturb the paths of each subset in a way that minimises the cost raised by re-planning their local interactions.

The intuition to this approach is that by viewing others in the crowd as moving obstacles, the planning of each individual can then be regarded as the problem of a single-person global navigation in a known dynamic environment. Since my technique maintains a set of global paths of every member, the answer to this problem is an one-against-others navigation strategy where a global path is planned against the complete trajectories of rest of the crowd. In my implementation, the global path planning is based on a motion graph query technique which supports full-body collision detection in goal-directed motion retrieval (Chapter 3). This allows for precise collision detection in planning close interactions, rather

Algorithm 1 Overview of incremental navigation planning

```
1: S ← INITIALISEPATHS()
2: loop
3:    I_all ← FINDINTERFEREDPATHS(S)
4:    if I_all = ∅ then
5:       return S
6:    for all I ∈ I_all do
7:       PERTURBPATHS(I)
```

than avoiding collisions based on instantaneous movements and coarse collision models. In addition, since each path is planned through motion graph queries, the resulting animation is smooth and natural-looking since it is concatenated using numerically similar human motion capture data.

My crowd navigation planning technique works by making the assumption that there exists a collision-free solution for each interference, and the perturbation to resolve that is small. Indeed, once paths interfere with each other, they become interdependent, and it is impossible to prioritise one path without compromising others. To address this, I take a snapshot of the trajectories before re-planning and use this snapshot to re-plan for global collision-free solutions independently. Because I am able to plan interactions very closely, I can assume that the difference between the paths of the snapshot and of the re-planning are small, and will become smaller and smaller over the iteration until the equilibrium is reached. Similarly, since my planning can react to collisions at a close distance, I can assume that perturbations do not disturb the paths before their interferences, and hence no backward propagation of collision would occur. I will describe how these assumptions are upheld in Section 4.4.

4.3 Implementation

To realise my navigation planning technique, I model the crowd as a set of global paths and design an incremental optimisation algorithm to resolve collision amongst them. This algorithm identifies the spatio-temporal interferences between paths in order to select and re-plan interfered paths without implicating others. Let S be a set of global paths representing individuals of the crowd such that $S = \{x_1, ..., x_n \mid x_i \text{ is a global path}, n \text{ is the size of the crowd}\}$. This framework is outlined as Algorithm 1 and can be summarised as three major steps:

- **Path initialisation (line 1).** The crowd is initialised by planning a path for each individual x_i regardless the rest of the crowd ($S - \{x_i\}$).

- **Path interference (Line 3).** Find all subsets $I, I \subseteq S$, each of which contains the individuals involved in a collision incident.

- **Path perturbation (Line 7).** Each collision incident is revolved by re-planning each member $\bar{x}$ in the subset I against $(I - \bar{x}) \cup (S - I)$ for a joint global solution.

The algorithm continues the interference step and the perturbation step iteratively until no further collision to be resolved (Line 5).

This section will first describe how I initialise paths through underlying query-based planning technique. I will then explain how I isolate interfered paths and select them in groups in Section 4.3.2. The joint optimisation of path perturbations is described in Section 4.3.3.

4.3.1 Path Initialisation

To initialise the crowd, I apply the environment-aware motion graphs (EMG) presented in Chapter 3 as the path planning technique to find each individual a global path. Given a starting location and a goal, the EMG technique finds a path as a sequence of motions that travels between these two locations. This motion sequence can then be derived into a trajectory that indicates the global locations and the corresponding poses over time.

To query a desired path, EMG relies on a cost function to evaluate the motion sequence in respect of certain constraints such as goal deviation and collision errors. Recall the cost function in Equation 3.4 which is used to find an optimal path across a set of static obstacles with minimum travelling time and distance. Given an individual's path $\bar{x}$ and a goal location q, the cost is calculated by

$$\mathrm{C}_{\text{initialise}}(\bar{x}) = \underbrace{\psi}_{\text{collision}} + \underbrace{\frac{|p-q|}{\Delta}}_{\text{deviation}} + \underbrace{max(\frac{\lambda}{\Lambda}, 1)}_{\text{distance}} + \underbrace{max(\frac{\tau}{\mathrm{T}}, 1)}_{\text{duration}}. \tag{4.1}$$

I use $\mathrm{C}_{\text{initialise}}(x)$ to generate paths for every crowd member and assume this solution to be their individual optimal paths.

Whilst it is less likely that this path initialisation will be collision-free, this approximation provides an insight to the flows and distribution of the crowd motions. Thus, we can regard this initial approximation as a "target" solution, and the aim of the incremental optimisation is to find a collision-free solution close to this solution.

4.3.2 Path Interference

The incremental optimisation begins by globally identifying every pair of interfered paths between crowd members. To resolve a collision, I include not only the collided individuals but also the proximate neighbours to be involved in the perturbation stage. Meanwhile, I need to keep the participating members to a minimum to reduce the computational load of the motion graph queries in path re-planning. I thus define a *vicinity radius* to draw a circle from the centre of the incident point and select the members inside this circle as a set $I_{\text{local}} = \{x_1, x_2, x_3, ... \mid x_i \in S\}$ of this local incident group. Figure 4.1(a) illustrates a collision incident occurs between red and green members in a cropped region of a initialised crowd. In Figure 4.1(b) we draw a circle using r_{vicinity} to include the blue one into this incident group for re-planning.

Because the way I initialise the crowd, often there exist multiple collisions among the crowd and one member might be involved in more than one local incidents. Since each path is initialised with a global

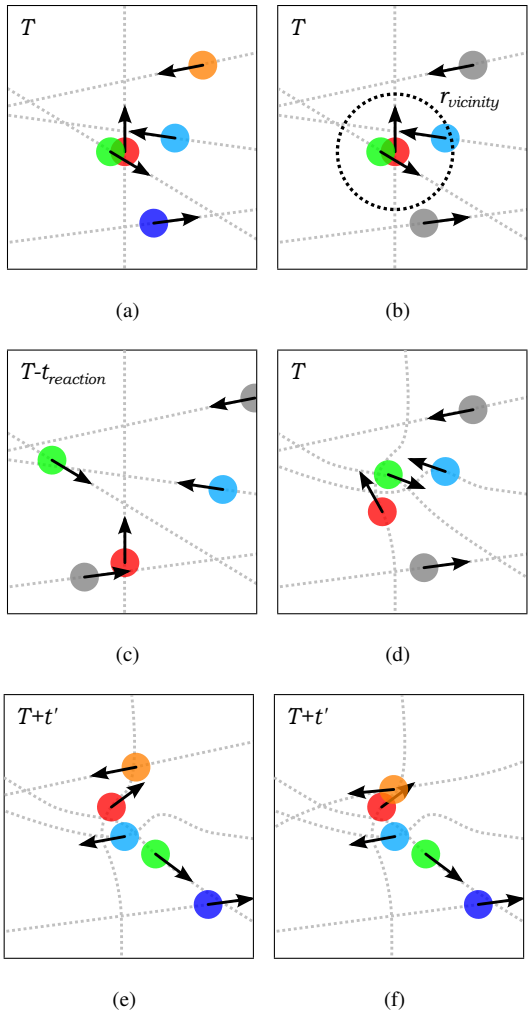


Figure 4.1: A cropped region of a collision incident resolution during mutual avoidance optimisation: (a) Initial paths. (b) Local incident detection in path interference. (c) Rollback mechanism. (d) Path re-planning in path perturbation. The result is either (e) collision-free, or (f) collided with another re-planning at $T + t', t' \geq 1$.

solution, a member that is re-planned due to the first incident encountered will have consequences on the following incidents. That is, it will be no longer involved in the following incidents directly. Therefore, I only resolve incidents of which none of the members is being re-planned prior to this incident. I define such incidents as *primal local incidents*. To find all primal local incidents in the current crowd solution, I first arrange each local incident in a queue that is prioritised with their temporal orders. I then scan through the queue and drop any local incident from the queue if any of its members is involved in an earlier incident. Let $I_{\text{all}} = \{I_1, I_2, I_3, ... \mid I \subseteq S\}$ be the priority queue and let I_i be the current local incident in the queue. I drop I_i from I_{all} if

$$I_i \bigcap_{I \in I_1 \text{to} I_{i-1}} I \neq \varnothing. \tag{4.2}$$

In other words, I ensure that all I in I_{all} are pairwise disjoint subsets. Therefore, re-planning of an earlier incident does not affect that of a later one in the same iteration while any unresolved incidents

Algorithm 2 Path Perturbation

Require: $I = \{x_1, x_2, x_3, ...\}$

1: **repeat**
2: $\hat{I} \leftarrow I$
3: $I' \leftarrow \{x' \mid x' \leftarrow \text{RePlan}(x, (\hat{I} - \{x\}) \cup (S - \hat{I})), x \in I\}$
4: $I_{\text{combo}} \leftarrow \text{MixAndMatch}(I, I')$
5: $I \leftarrow \arg\min \{\text{CombinedCost}(I_i) : I_i \in I_{\text{combo}}\}$
6: **until** $I = \hat{I}$
7: **return** I

are to be dealt with at a future iteration. Another benefit is that we would be able to resolve these primal local incidents in parallel, although it is not implemented in our experiments. I build this queue at the beginning of each iteration to discover the remaining primal local incidents and resolve them individually.

4.3.3 Path Perturbation

Having isolated primal local incidents, we can now begin to perturb interfered paths by re-planning their interactions. The path perturbation itself is a 4-step iterative optimisation as outlined in Algorithm 2. At the first step, I take a snapshot of the paths in I to form a new set $\hat{I}$ (Line 2). I then re-plan each path in I against the trajectories of other members from the snapshot for an alternative global path to form another set I' (Line 3). Then at the next step, I mix and match the re-planned and original paths to enumerate all possible combinations I_{combo} (Line 4). Finally, I find the combination in I_{combo} which minimises the combined cost including the their collision errors. I repeat these steps until no alternative can be found to improve the situation (Line 6).

One key to my global navigation approach is the that the global path planning technique, the EMG method, is able to exploit the most feasible motions to avoid obstacle using precise collision detection. From an individual's point of view, other members in the crowd can be regarded as dynamic obstacles. To enable dynamic-obstacle collision avoidance in EMG, I find all collisions between $\bar{x}$ and the trajectories of the snapshot of other members of the crowd. I then count the total number of collided poses in $\bar{x}$ as an additional term ϕ to Equation 4.1 as a new cost function:

$$\mathrm{C}_{\text{against}}(\bar{x}) = \underbrace{\overbrace{\phi}^{\text{dynamic}} + \overbrace{\psi}^{\text{static}}}_{\text{collision}} + \underbrace{\frac{|p-q|}{\Delta}}_{\text{deviation}} + \underbrace{max(\frac{\lambda}{\Lambda}, 1)}_{\text{distance}} + \underbrace{max(\frac{\tau}{\mathrm{T}}, 1)}_{\text{duration}}. \tag{4.3}$$

Importantly for my incremental method I can also partially re-plan a path before the incident point to minimise the implication to the current crowd solution. I define a *reaction time* to determine how far back we wish to begin to react to the collision. Let $\bar{x} = (m_1, m_2, m_3, ...)$ be the motion clips of path $\bar{x}$ and let T be the time of the first collision incident. I find clip m_i that contains the pose at $T - t_{\text{reaction}}$ and only replace sub-paths of $\bar{x}$ from the end of the previous clip m_{i-1} when optimising a path. Note

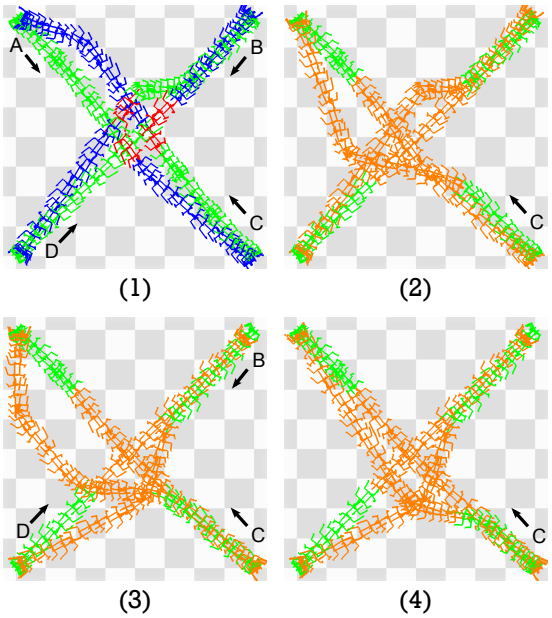


Figure 4.2: Iterations of path re-planning during four-person crossing.

that since I am able to avoid obstacles very closely, the re-planning only requires a short reaction time to find an alternative path. Thus, I find reaction time to be a small value such that $t_{\text{reaction}} \leq \frac{r_{\text{vicinity}}}{V_{\text{max}}}$, where V_{max} is the maximum moving speed found in the motion database, to minimise the disturbance of the re-plans.

Here I use an example to visualise the path perturbation optimisation. Suppose we have an incident group of four members $I = \{A, B, C, D\}$. Each of them is moving diagonally from one corner to the opposite in a square. I begin with an iteration shown in Figure 4.2(1) where red poses indicate the collision incident and the green ones and blue ones represent the animation before and after the incident, respectively. To perturb the paths, I roll back each individual x_i and partially re-plan against the complete history of the snapshot of other members in the incident $(\hat{I} - \{x_i\})$ and the rest of the crowd $(S - \hat{I})$. This allows us to always consider the global intensity and movement of the crowd. I hence re-plan to obtain a new set $I' = \{A', B', C', D'\}$. I then mix and match I and I' by enumerating the power set of I' and combine with their complementary members in I as new sets:

$$I_{\text{combo}} = \left\{ \begin{array}{c} \{A, B, C, D\} \\ \{A', B, C, D\} \\ \{A, B', C, D\} \\ \vdots \\ \{A', B', C', D'\} \end{array} \right\}. \tag{4.4}$$

To evaluate each combination, I denote Φ to be the number of the collisions occurred between these members as *local collision errors*. I discard the collision error ψ in Equation 4.1 for individual costs and summarise with the local collision error Φ. Therefore, I find the best combination by

$$\bar{I} = \underset{I \in I_{\text{combo}}}{\arg\min}\, \Phi + \sum_{x \in I} \left(\frac{|p - q|}{\Delta} + max(\frac{\lambda}{\Lambda}, 1) + max(\frac{\tau}{\mathrm{T}}, 1) \right). \tag{4.5}$$

Figure 4.2(2) shows the second iteration after I re-plan the paths (coloured in orange). Although I re-plan each individual to propose a solution, not every re-plan is accepted unless the combined cost is reduced. Here at iteration (2) only C is accepted and updated to the crowd. The optimisation continues when any further improvement is found. At iteration (3), B, C and D all re-plan the paths successfully due the change made by C in iteration (2). For instance, D finds a new path which arrives at the goal more closely. C is also able to re-plan a shorter path in iteration (4) that minimises the travel cost. I enumerate all possible combinations of available proposals (successful re-plans) to find the best combination and update to the crowd. In this case, multiple mutual changes create better overall avoidance interactions, such as overtaking someone and giving way to the others, than an individual improvement.

4.3.4 Refinements

We should now obtain a set of global paths without any interferences if every perturbation is successful and found a collision-free solution. However, since I always re-plan each individual against the trajectories of the rest of the crowd, there are occasions where an individual is avoiding someone that is no longer using the same path. This is because those trajectories are the snapshot of the previous iteration and some of the paths might also be re-planned simultaneously. I address this by a secondary re-planning to refine each path in accordance with the current crowd movements. I prioritise the paths according to their individual costs such that the potential sub-optimal paths are re-planned first. I update each successful re-plan to the crowd before the next re-plan so that the movement of the entire crowd is always up-to-date. Note that a re-plan is only accepted if the cost of the new path is lower. Therefore the secondary re-planning can only refine those paths that can be further improved without compromising others.

4.4 Non-degeneracy and Completeness

Fundamentally the algorithm uses an optimisation step that perturbs the interfered paths locally in order to resolve a collision. Hence, the key to determine whether the algorithm completes the search is the existence of a collision-free local solution. Indeed, one can always construct pathological examples for which no solution exists. In my experiments, I have found that the local optimisation does produce a collision-free solution. If the local optimisation does produce a collision-free path, the global search is guaranteed to terminate. Here, I revisit the example illustrated Figure 4.1 to outline the proof in two stages as follows.

- **Stage 1:** Assume we have a collision at frame T and it is the only incident at this time as illustrated in Figure 4.1(b). Note that an incident includes a conservative estimate of those members that will be involved in the collision by taking all members within r_{vicinity} of each other. This splits the crowd into two sets S_{coll} (coloured), S_{free} (grey).

  The rollback mechanism would then roll everyone back to $T - t_{\text{reaction}}$ (Figure 4.1(c)). The optimisation is asked to find a collision-free path for all members in set S_{coll} from $T - t_{\text{reaction}}$ but it also

uses the total history of S_{free} as moving obstacles. This optimisation should return a collision-free path to the end of the animation ($T_{\text{finish}}, T_{\text{finish}} > T$) such as the result shown in Figure 4.1(d). It can fail, in which case we ignore the collision. It does not fail in our examples. A more conservative view is that it generates a collision-free route until $T + 1$. In which case the optimisation progresses at least one frame, and any new added collision is one frame in the future.

- **Stage 2:** In the general situation we have multiple collisions at T. But each of these collisions is separated by at least r_{vicinity} and that radius is much larger than any crowd member can travel in one frame. Assume we have two collisions, the generalisation is easy by induction, and we have sets S_{collA}, S_{collB}, S_{free}. We solve for S_{collA}, but the optimisation uses S_{collB} and S_{free} as moving obstacles. Following the argument for Stage 1, assuming the optimisation works, there can be no new collision involving S_{collA} until at least $T+1$. Further, it is impossible for any member in S_{collA} to collide with any member in S_{collB} at T because the two groups are further apart than r_{vicinity}. Thus solving any one collision at T cannot make the existing problem more complex through changing the members involved in the incident. Thus we make progress at T and can then solve the collision for S_{collB} at which point the optimisation progresses to at least $T + 1$.

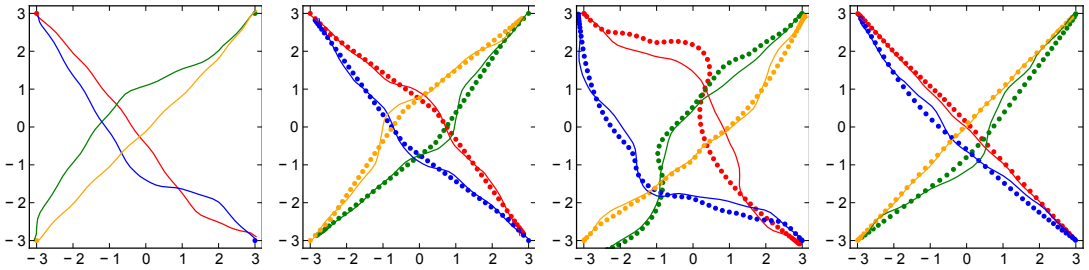
4.5 Results

I have generated test crowd animations that compare our incremental navigation planning method with comparable crowd dynamic simulations. The algorithm is implemented as Python interpreter modules with core procedures written using the C++ language and the resulting animations are visualised in the Maya animation software. The underlying path planning technique, the EMG method, is effective to find long full-body animations (more than 30 seconds) that avoid crowd trajectories and static obstacles and arrive at destinations. For comparable methods, I apply the path following technique in Section 3.3.1 to approximate motions on the trajectories of their simulations since they do not generate full-body animations. I use the same "walking" motion graph built in the previous chapter (see Table 3.1) to plan paths and generate motions.

Here, I am interested in a list of criteria to evaluate the effectiveness of a crowd animation:

- Is the result collision-free?
- Does each member of the crowd arrive at the goal?
- How long does it take for every crowd member to reach its goal?
- How frequently do the crowd members change their courses and paces in order to avoid each other?

To determine whether there are collisions in the final animation, I perform full-body collision detection on the resulting animation for every member pair. My method should not produce collisions, but



(a) PIP (b) RVO2 (c) OpenSteer (d) Pettré et al.

Figure 4.3: A comparison between simulated trajectories (dots) and fitted motions (solid lines) in the four-person crossing test scene. My PIP method does not require to approximate a trajectory in order to generate full-body animation.

for the comparable methods we have to fit motion to the paths they generate, and in this process collisions might be generated (see below). The duration is timed from the beginning of the simulation until every member reaches its destination. Finally, I calculate and sum ground trajectories of each member to determine the total travel distance of the crowd. I also highlight the total deviation between position of the end of individual animations and their goal locations. I calculate the speed profile to compare the frequency of behaviour changes between the crowds.

4.5.1 Four-person Crossing

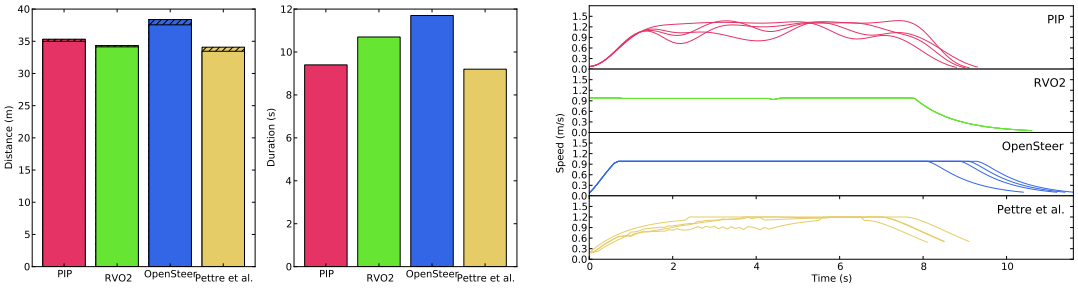
I first designed a simple test scene using a small crowd of four crossing from four corners of a square diagonally to the opposite corners. I use this initial condition to generate an animation using my method, referred as PIP (Path Interference Perturbation), and compare to the ones generated by local models. I chose OpenSteer (Reynolds, 1999) and RVO2 (Snape et al., 2010) to be our controls since they are considered effective on simulation avoidance behaviours and have been applied on various games and graphics applications. I also compare the results with the collision-avoidance model proposed by Pettré et al. (2009) as it reproduces the real human avoidances by adapting captured pair-interaction motion data. For OpenSteer and RVO2 I use the maximum radius of the bounding cylinder found in the walking motion database, which is $42.7cm$, as a constant radius for each agent. Pettré et al.'s model does not define a bounding radius as they only rely on instantaneous mutual distances and bearings as well as timing conditions during an interaction. I also apply the maximum and averaged walking speed calculated from the database ($1.63m/s$ and $0.98m/s$) as the agent's maximum and desired speed, respectively.

Note that since these methods only simulate the trajectories of the crowd, we need to fit full-body animations to approximate them. I first attempt to search the motion graph with the simulated trajectories as way-points. Despite my best efforts, I failed to find extract motions that follows the trajectories for the three controls. Figure 4.3 shows the deviation between simulated trajectories (dots) and the ground trajectories of the root position (solid lines).

Failure to follow these planned trajectories are likely to yield collisions in a more congested crowd. To work around this, I additionally enforce the trajectory following by fixing the root position of each pose distributed along the curve formed by the trajectory. Due to mismatched root speed between the

	Number of collisions					
	4-person crossing		Multiple crossing		Central locking	
PIP	0		0		0	
Root-fixing	Before	After	Before	After	Before	After
RVO2	0	0	371	4	1073	2
OpenSteer	50	0	402	56	287	6
Pettré et al.	8	0	–	–	–	–

Table 4.1: Collision errors before and after root-fixing.



(a) Travel distance and duration comparisons.

(b) Speed profile comparisons.

Figure 4.4: The results of the four-person crossing crowds. (a) While PIP chooses slightly longer routes than RVO2 and Pettré et al., the crowd of PIP arrives at the goals almost as soon as Pettré et al.'s. (b) The speed profile of PIP is smooth and uses a similar spectrum of speed as Pettré et al.'s pedestrian model.

trajectory and animation, fixing the root position creates noticeable visual artefacts such as foot skating as can be seen in the supplementary video (Appendix A.2). However, due to the non-linear pelvis movement of human walking motions, there is no simple way to avoid foot skating when fixing the root position. I report both the fixed and unfixed collisions in Table 4.1.

Figure 4.4(a) shows the measures of the travel distances and durations from the four-person crossing crowds. The hatched areas in the distance measurements indicate the cumulative deviations to the goal locations, showing all crowds are able to arrive closely to the goal. Note that OpenSteer requires a longer animation to achieve the goals compared to RVO2, Pettré et al. and my PIP method. This is because the individuals in OpenSteer repels each other in order to avoid collisions, where as the other crowds can avoid each other relatively more closely. Although PIP chooses the paths that are slightly longer than RVO2 and Pettré et al.'s model, the crowd of PIP arrive at the goals as soon as Pettré et al.'s. Figure 4.4(b) compares the speed profile measured in the trajectories of the four crowds. Both RVO2 and OpenSteer are using a lower speed (around $0.9m/s$) to avoid collisions. PIP is able to avoid collisions using a higher speed, with a similar speed spectrum to Pettré et al.'s model.

From the supplementary video A.2 we can clearly see that although RVO2 can resolve collision very closely, it creates congestion in the centre of the scene that does not look plausible. One key finding of Pettré et al.'s work is that the cooperative behaviours are crucial to simulate human-like avoidance.

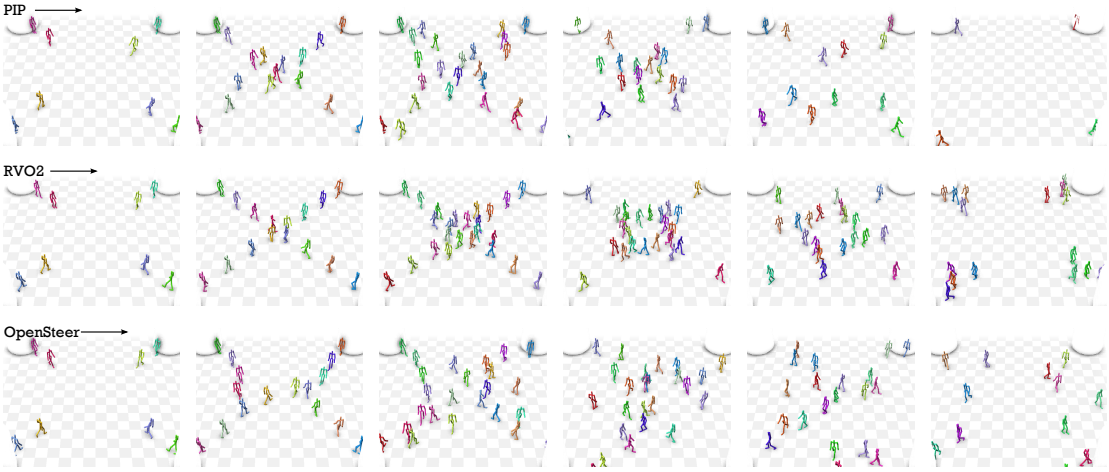
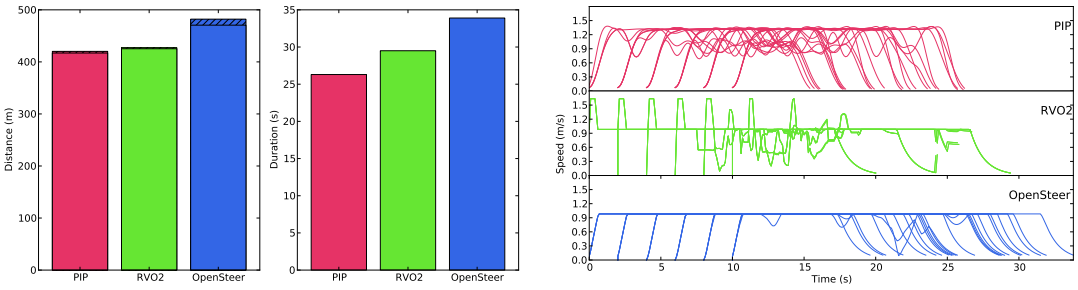


Figure 4.5: Snapshots of the resulting animations for the multiple crossing crowds.



(a) Travel distance and duration comparisons.

(b) Speed profile comparisons.

Figure 4.6: The results of the multiple crossing crowds. (a) PIP plans shorter routes and animations that arrive at the goals sooner. (b) RVO2 shows sudden changes of speed indicating anomalous motions.

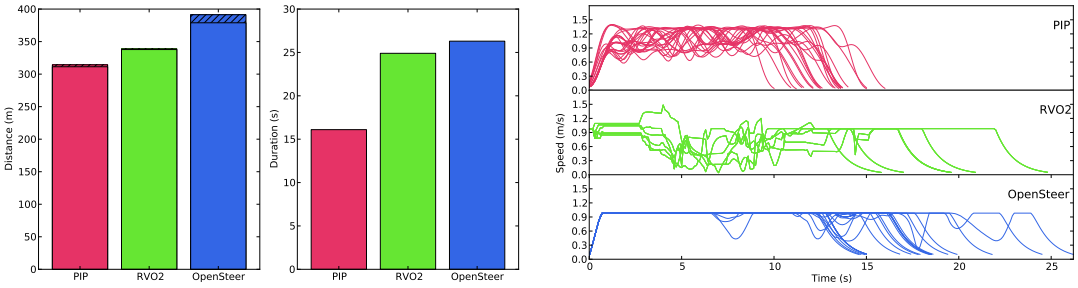
However, this model does not consider any collision models but only rely on the instantaneous movements of the neighbours to avoid collisions. As a result, they can still cause collisions when individuals are proximate. Notice that in the video in my method the blue person and the red person give way to the others by slowing down and changing course, respectively. The result demonstrates that my method is capable of generating similar cooperative behaviours of human interactions without additional interaction data. Also, since my collision avoidance is planned using full-body collision detection, my result can avoid each other very closely without causing collisions.

For the subsequent test scenes, we can no longer consider Pettré et al.'s method as it only focuses on avoidance behaviour and does not guarantee the planning of a collision-free animation.

4.5.2 Multiple Crossing

Figure 4.5 shows another example of 4-way crossing but now with streams of crowd members in each direction. Every 2 seconds four new members enter the scene. This happens 6 times for a total of 24 crowd members. I designed this test scene to determine how each algorithm resolves the congestions

Figure 4.7: Snapshots of the resulting animations for the central locking crowds.



(a) Travel distances and duration comparisons.

(b) Speed profile comparisons.

Figure 4.8: The results of the central locking crowds. (a) OpenSteer shows larger goal deviation (hatched areas), indicating the failure to arrive close to the goals due to the repulsive collision avoidance strategy. (b) Both RVO and OpenSteer show drastic speed changes while the crowd of PIP can avoid each other with a smoother and faster speed profile.

in the middle as the population increases. Before root-fixing, both OpenSteer and RVO2 deviate from the simulated trajectories significantly. Table 4.1 shows the number of frames that contain a collision between each pair before and after root-fixing. In order to avoid each other, OpenSteer maintains a large clearance between individuals resulting in a more scattered crowd where crowd members generally take longer routes, as can be seen in Figure 4.6(a). Without planning, the crowd members of RVO2 all choose the shortest path. As a result, they slow down significantly and create heavy congestion in the middle, causing constant anomalies such as sudden speed and directional changes. This can also be seen from the speed profile comparisons in Figure 4.6(b), where individual speeds increase and decrease drastically. My crowd remains aggregated and yet flowing freely and smoothly without frequent pace or course changes.

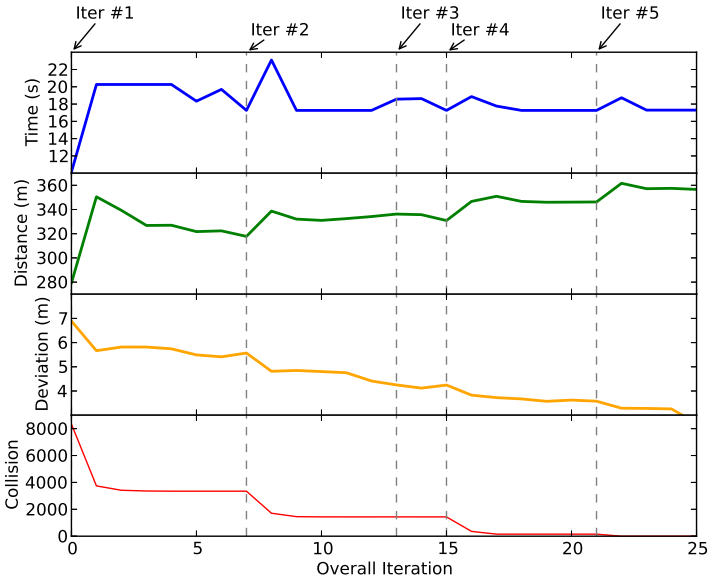


Figure 4.9: The convergence of the costs over the incremental optimisation steps of the central locking simulation with our algorithm. Notice that the algorithm trade-off travel time and travel distances for minimising goal deviations and collision errors over iterations.

4.5.3 Central Locking

Figure 4.7 shows another test scene with 24 individuals moving through the centre of a circle to the opposite side. OpenSteer produces unnatural large swirls where each crowd members starts by heading around the perimeter of the circle. The crowd members also find it difficult to arrive closely to the goals due to the repulsive force required to avoid each other. This is shown in Figure 4.8(a) where OpenSteer not only requires longer routes and longer time to reach the goals, but also creates a large goal deviation (the hatched area) compare to the other crowds. Crowd members in the RVO2 simulation clump together in the middle causing sudden directional and speed changes. My crowd is able to generate the most feasible motions that move along a plausible route with a much smoother speed profile, as can be seen in the supplementary video A.2 and in Figure 4.8(b).

Figure 4.9 shows the progress of four costs measured during the optimisation of the central locking simulation using the PIP algorithm. In this figure, each grey dash line indicates a new iteration of a single independent incident. Since we begin with a crowd by planning each member individually, the initialisation creates high collision errors as everyone is colliding in the centre of the circle. Throughout the optimisation, collisions are reduced significantly at the beginning of each iteration by increasing travel distance and duration costs, such as changing the course with a longer route and/or slowing down the pace to make way for others. Both of these travel costs as well as the arrival deviation are then decreased over the mutual avoidance optimisation until the beginning of the next iteration. I have observed similar convergence behaviour in all other crowd animations using my method.

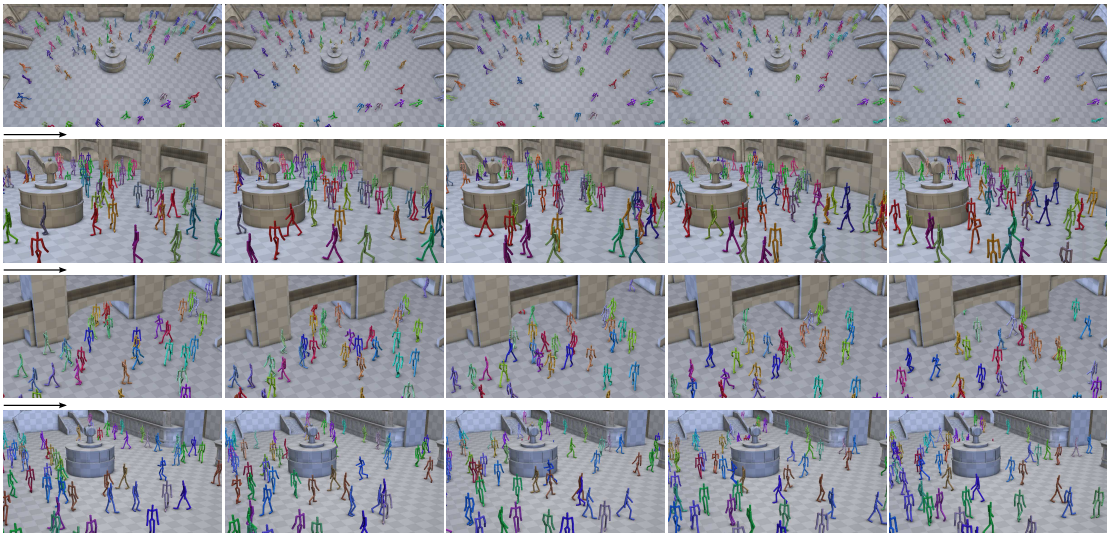


Figure 4.10: Snapshots of the resulting animations for the train station scene.

CMU trial no.	Description
16_08	run/jog, sudden stop
16_36	run/jog (slow jog)
16_38	run/jog, veer left
16_40	run/jog, veer right
16_42	run/jog, 90-degree left turn
16_44	run/jog, 90-degree right turn
16_46	run/jog (fast run)
16_56	run/jog (from start)

Table 4.2: The motion capture data used to construct the "running" motion graph.

4.5.4 Station Concourse

Finally, I design a concourse in a train station scene to demonstrate that my method can scale to a mid-sized crowd (Figure 4.10). In this test scene, 100 individuals are moving across the concourse to the other side. I also include environmental obstacles which are modelled as static primitive geometries such as cubes, spheres and cylinders. As well as the "walking" motion graph, another "running" motion graph (Table 4.2) is also used in this test scene. While it is considerably dense, all crowd members, including the running ones, do not need to slow down to avoid each other. Please refer to the supplementary video A.2 for the resulting animations.

4.6 Discussion

In this chapter, I have presented a navigation planning technique that generates optimal paths and free-flowing animations for many individuals. My technique unifies global path planning with interpersonal collision avoidance. It is thus able to find global solutions that move individuals towards their goals in soonest possible routes, with minimum obstruction to avoid each other.

My technique is novel in that it enables dynamic-obstacle collision avoidance in goal navigation via a multi-path interference-perturbation minimisation. This allows for per-path global planning against the evolution of the known trajectories of others in the crowd. The incremental optimisation framework guarantees that I resolve path interferences in respect of their spatio-temporal relationships such that all collisions will eventually be fixed. One key reason for this is that the underlying query-based planning technique allows for precise collision detection, and hence I can plan close interactions between individuals. Also, because my method does motion planning and motion synthesis simultaneously unlike other techniques, I do not need to fit locomotion animations to the resulting trajectories. Thus I avoid a step that would otherwise potentially introduce anomalous behaviours and visual artefacts.

This navigation planning begins with a crowd of which every member is choosing a desired route in respect of certain high-level constraints such as least travel time and distance, and the optimisation is looking for a collision-free solution close to this initialisation. This property allows the user to test the feasibility of a certain crowd configuration by simply specifying the starting and the goal locations. Furthermore, the user can choreograph the crowd by modifying or adding or removing a member in the crowd and re-start the optimisation and the algorithm will automatically find a new solution with minimal perturbation to the existing solution. Hence I believe this technique would be ideal for ensemble behaviour planning and analysis purposes.

My algorithm is guaranteed to find a global solution under the premise of the existence of non-degenerate local solutions. That is, there exists at least a local solution for each interference, and the perturbation to resolve that is small. I have shown that is these assumptions can be maintained in my experiments. However, to determine whether my algorithm would work under various conditions, such as different densities or flows of the crowds, further investigation is needed.

The aim of my crowd animation technique is different from conventional crowd simulations. I focus on global navigation planning rather than simulating crowd behaviours, and yet my results are plausible in that it based on the assumption that motion graph is a good representation of the known behaviour flows. Thus, individual animations are guaranteed to be smooth and natural-looking. It is worth noting that while it is able to include more motions with different paces into the motion graph, currently there is no explicit control over the speed of the collision-avoidance strategy. Such preferences might be addressed by searching within sub-graphs that contain motion with the same pace or incorporating as an additional term in the cost function. One might also argue that the uncertainty due to limited human perception is vital to the plausibility of crowd interactions, and hence my results will almost certainly not happen in the real-world scenario. One possible solution is to make my path planning “less global” by only considering crowd members in a smaller range, although it might increase the computation as it

might require more path perturbations along the way.

My technique is far from real-time, with the computation of the animations taking minutes for the simpler cases through to several hours for the animation of the crowd of 100 individuals in the congested train station. However the quality of the resulting animation is high with no collisions and consistent behaviour from the individual crowd members. My global path planning relies on the motion graph to query motion sequences to perform full-body collision detection. These are very computational intensive tasks. A possible solution is to replace the actual motion sequence as a series of speed and direction changes. In doing so we can simplify the motion graph to a much simpler representation to allow a coarser but faster path planning. It would be interesting to analyse the impacts when we trade-off precision for scalability.

Chapter 5

Plausibility Metrics

Having discussed new techniques for planning human motions for scene navigation and collision avoidance, this chapter considers a new benchmarking approach for evaluating the plausibility of navigation animations. In Chapter 3 I demonstrated that collision detection can be incorporated into motion queries to generate navigation animations that are both natural-looking and artifact-free. Indeed, at the motion-synthesis level, the motion graph offers a human-motion model that ensures structurally similar motions are concatenated with smooth transitions, and thus is able to guarantee the animations queried from the motion graph would look plausible. From the literature review we have also learned that various methods have been proposed to identify common motion synthesis anomalies such as motion glitches and footskates. However, despite there are various techniques that can find solutions that avoid obstacles and reach the goal, yet there is no quantitative approach to evaluate whether the resulting animations would look plausible from the motion-planning point of view.

One main interest of past studies on the evaluation of navigation animation is to detect and avoid anomalous patterns in the crowd. For instance, the composition of the crowd in still images are randomised to evaluate the impact on human perceptions (Ennis et al., 2008). Crowd formations are evaluated to determine the randomness of their patterns (Ju et al., 2010). Also, the selection of motions to construct crowd motions is evaluated to increase the variety (Pražák and O'Sullivan, 2011). Another approach is to exploit real-world human navigation data. For example, collision-avoiding animations are compared with captured human interactions (Pettré et al., 2009), but their main concern is analysing the reaction to the collision rather than the entire animation. Real-world scenarios are also reconstructed such that the simulated motions can be compared with the original pedestrian trajectories (Guy et al., 2012). However, the analysis of this approach is case-specific and hard to generalise to evaluate other types of behaviours.

In this chapter, I attempt to define and measure the "plausibility" of navigation animations. The aim is to identify anomalous behaviours in situations when travel through obstacles and avoid collisions, and thereby provides an insight towards generating anomaly-free navigation animations. I focus on the detection of common collision-avoidance anomalies, namely the erratic travel paths and motion changes in individual animations, as well as the unnatural interpersonal clearances and synchrony in crowd nav-

igation. To address this, I design a set of metrics to quantitatively assess individual trajectories and intra-group correlations to highlight potentially anomalous behaviours in navigation animations. The notion is that in order to generate animations that look plausible, individual movements must comply with the physical limits as well as the biomechanical characteristics of human motions. Also, to portray a plausible collision-avoidance behaviour between crowd individuals, the degree of mixture of interpersonal coordination and decision-making uncertainty appears to be crucial. By measuring the kinetic energy, curvilinear kinematics, pair-wise distances, and group synchronisation of individuals' trajectories, my metrics are able to compare between synthesised navigation animations and indicate their potential anomalous behaviours.

The remainder of this chapter is as follows: I first describe the anomalous navigation behaviours I wish to detect in Section 5.1. I then define the plausibility metrics and describe how they are implemented in Section 5.2, where I use the navigation animations produced in the previous chapters to demonstrate how motions are evaluated using these metrics. In section 5.3 I design experiments to discuss how different planning strategies affect the plausibilities of the results.

5.1 Anomalous Collision Avoidance Behaviours

To evaluate the plausibility of navigation animation, I propose to analyse the curvilinear motion of individual trajectories as well as their pairwise spatio-temporal correlations to detect anomalous behaviours. My main concern is to identify the motion anomalies caused by avoiding collisions. My observation is that most anomalous collision-avoidance behaviours are caused by inaccurate collision models and naïve avoidance policies, resulting sub-optimal animations such as erratic travel paths and excessive change of motions. These models and policies can also damage the collective plausibility of crowd navigation, creating anomalous behaviours such as the violation of interpersonal clearances and the formation of hyper-synchronised movements. To detect these anomalies, I focus on four quantitative attributes that can be measured from the trajectories:

- **Excessive energy consumption.** This is caused by sub-optimal animations such as choosing an erratic travel path or making constant pauses to avoid collisions. The result for this is an anomalous animation that does not appear to be heading towards the goal. It is suggested that humans tend to choose a path that requires least effort to the goal, and such principle can be modelled in the collision-avoidance strategy to generate emergent crowd behaviours (Guy et al., 2010). To evaluate excessive energy usage in navigation animations, I compute a minimum biomechanical effort for each individual and compare that with the estimated energy required by traversing the path.

- **Hyperactive and impulsive change of motion.** These unnatural behaviours are caused by disregarding the biomechanical characteristics of human movement when generating navigation animations. This is because most techniques only consider the upper limit of the velocity when avoiding collisions. This results in hyperactive and impulsive motion changes when the crowd is dense and individuals' paths are crossing each other's. There has been a discussion on the evaluation

of compressed human motions (Arikan, 2006), and the study suggests that humans tend to notice high-frequency errors as it actually requires more energy to make such movements. To highlight potential motion change anomalies, I measure the rate of change of acceleration in the navigation animations and determine how they are diverged from the captured human data.

- **Inaccurate interpersonal clearances.** This anomaly is mainly due to inaccurate collision detection models and avoidance strategies. For example, using a strong repulsive force to avoid collisions might cause excessive clearances and erratic oscillating motions between individuals. To indicate interpersonal clearances, I measure the nearest distances between individuals to estimate their minimum collision boundaries in order to highlight possible interpersonal collisions and evaluate the accuracy of the collision models.

- **Hypersynchronised group movements.** Such behaviour occurs when individuals converge and begin to avoid each other using the same rules. As a result, they mirror each other's movement and form a symmetrical formation. Although intra-group interactions such as interpersonal collision avoidance requires a certain level of local coordination, it is unlikely individuals would respond to the collision with identical reactions in an analogous real-world scenario. I measure the acceleration correlation between individuals to determine their group synchrony.

It is worth noting that these metrics are based on the assumption that individual animations are generated to portray the intention to reach their destinations. Personal preferences, such as the tendency of avoiding crowded regions or choosing certain speeds, are not considered in these metrics.

5.2 Implementation

The plausibility metrics rely on the analysis of the curvilinear motion of individual trajectories and their pairwise spatio-temporal relationships. To describe how these metrics are implemented, I apply them to results from the multi-character navigation experiments in Chapter 4. There are three test scenes in the multi-character experiments: "four-person crossing", "multiple crossing", and "central locking". For each of the experiment, I apply three different collision-avoidance techniques to generate navigation animation. These techniques are "Path Interference Perturbation (PIP)" (Chapter 4), "RVO2" (Snape et al., 2010), and "OpenSteer" (Reynolds, 1999). Each trajectory of the results generated by each method consists of a series of 2-D coordinates indicating the ground locations at each timestep (10 frames per second). For metrics that requires benchmark human motion, I perform the measurements in the "walking motion" and the "running motion" database described in Chapter 3 and 4.

5.2.1 Resistance

To compare the energy used to avoid collisions, I define a hypothetical *ideal path* for each individual and calculate the ratio of the energy difference between the actual and ideal path, to the energy of the ideal path. An ideal path is the straight line between the initial location to the goal of an actual path,

and one can travel along this path using a desired speed by disregarding any obstacles en route. Thus, the energy difference between the actual and ideal path can be regarded as the "resistance" caused by collision avoidance. Thus, the resistance ratio is defined as

$$Resistance = \frac{E_{\text{actual}} - E_{\text{ideal}} + E_{\text{deviation}}}{E_{\text{ideal}}}, \tag{5.1}$$

where $E_{\text{deviation}}$ is an additional energy to penalise a path that does not arrive close to the goal.

Given a path of a navigation animation, the energy is estimated by calculating the velocities along the trajectory. My energy estimation is based on the least-effort model by Guy et al. (2010). Their study shows that the energy usage for a human travelling along a path Π can be estimated by

$$E(\Pi) = m \int_{\Pi} (e_w |v^2| + e_s)\, \mathrm{d}t, \tag{5.2}$$

where v is the instantaneous velocity at a given time-step t, m is the body mass of the human, and e_w and e_s are the biomechanical coefficient and constant to determine the energy consumption rate for moving and standing still, respectively. They also find that based on this model, one can traverse the path with least energy using an optimal moving speed, which is $1.33\ m/s$ for an male adult. Therefore, given the travel distance of a path, one can obtain the minimum energy by applying the optimal speed as a constant velocity to traverse this path. The indication of this energy estimation is twofold: first, the energy consumption is proportional to both travel time and travel distance, and second, constantly choosing a speed that is too low or too high will cause the energy to increase. These situations all indicate signs of implausible human navigation behaviours.

To measure the resistance, I first estimate energy E_{actual} for the (actual) path. I then estimate minimum energy E_{ideal} and $E_{\text{deviation}}$ by calculating the linear distances of the ideal path and the deviation between the end of the path and the goal, respectively. The resistance is obtained as described in Equation 5.1. The smaller the resistance is, the less excessive energy required to avoid collisions, and hence more plausible. Note that the minimum resistance will be slightly greater than zero because the necessary energy to avoid collision.

Figure 5.1 shows the resistance ratio measured in the multi-character navigation experiments. Each block is split into the ratio of the energy difference between the actual and ideal path (solid) and the ratio of the penalty energy caused by the deviation atop (halftone). Note that for each experiment the ideal paths are the same across three methods as they share the same initial and goal locations. The measurement indicates that PIP maintains lower resistance ratios in more complicated scenes such as "multiple crossing" and "central locking" while OpenSteer requires most energy in all three cases. The result also shows that the resistance increases as the situation becomes more congested. For example, in "central locking", 24 individuals are moving towards each other at the same time and thus each individual would need to negotiate his or her way through with more obstacles than the other two experiments. Finally, while PIP can arrive at the goals very closely, both RVO2 and OpenSteer show signs of difficulty is arriving close to the goals.

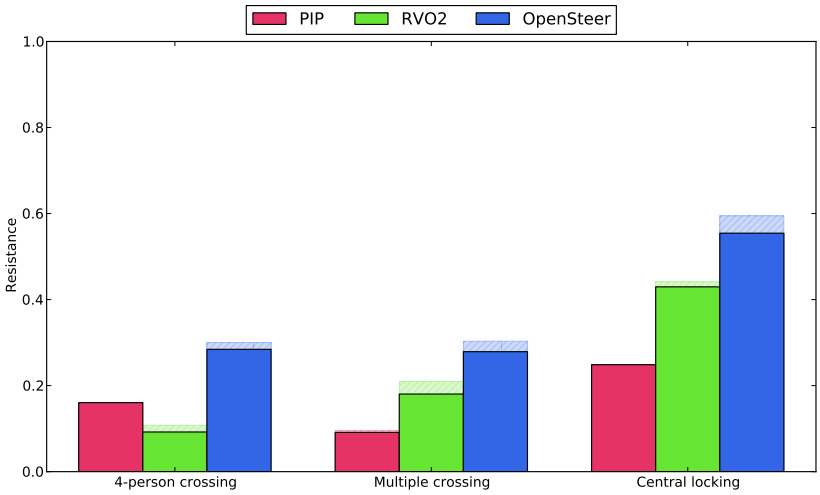
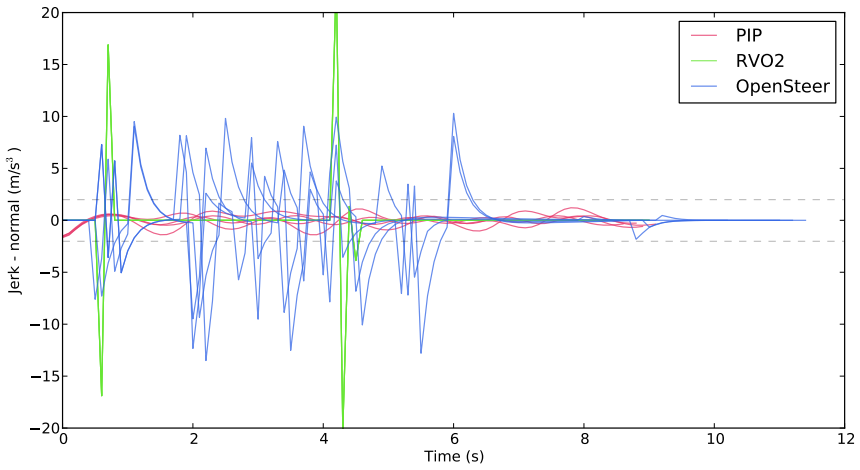


Figure 5.1: Benchmark results for energy consumption. The resistance ratio of the energy used to avoid collisions are compared to indicate possible excess energy in the navigation animations.

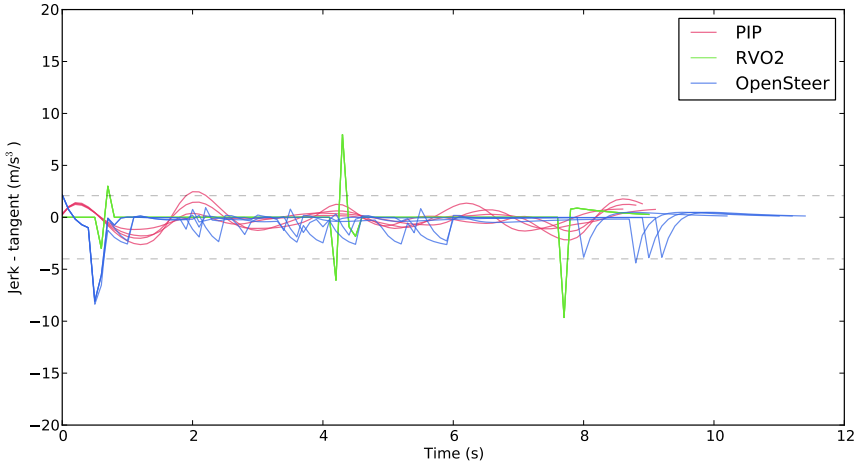
5.2.2 Jerkiness

To detect anomalies in behaviour changes, I examine the rate of change of acceleration, known as *jerk* (m/s^3), in navigation animation and compare against the same measurements in captured motion data. I define the "jerkiness" of a path to be the standard deviation measured from observing the first derivative of the acceleration over the trajectory. Therefore, a path containing frequent and sudden changes of acceleration will show high data variation and is hence jerkier. To measure the change of behaviours more accurately, I consider the jerkiness of the turning behaviour and pace-change behaviour separately by analysing the corresponding centripetal and tangential accelerations of the curvilinear motions. I examine the jerkiness measured from the locomotion database and apply them as benchmarks to indicate anomalous behaviour changes in the navigation animations.

Given a path of a navigation animation, the jerkiness is measured by calculating the first derivative of the acceleration along the trajectory and measure the standard deviation of the data. To distinguish turning behaviours from pace-changing behaviours, I examine the acceleration of the directional and tangential component respectively using the normal-tangential coordinates (nt-coordinates) of the curvilinear motion of the path. Figure 5.2 shows an example of the normal and tangential jerk measured in the "four-person crossing" experiment. Each graph is accompanied by two horizontal dashed lines indicating the upper and lower limit of the corresponding jerk measured from the "walking" database. Both RVO2 and OpenSteer show high frequency of acceleration changes. Also, the amplitude of their acceleration changes are relatively large and exceed the upper and lower limits occasionally. In contrast, paths created by PIP show smooth and stable acceleration changes, ranging within or near the upper and lower limits. This is because the paths of PIP are generated directly using the motion database. By measuring the standard deviation of the change rate, we can estimate the frequency and amplitude of the acceleration changes of a path and determine how they diverge from that of the motion database.



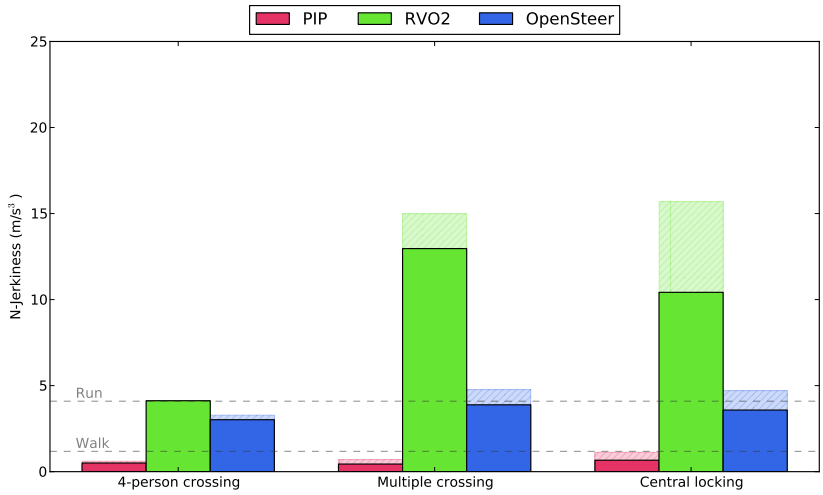
(a) Rate of change of normal acceleration.



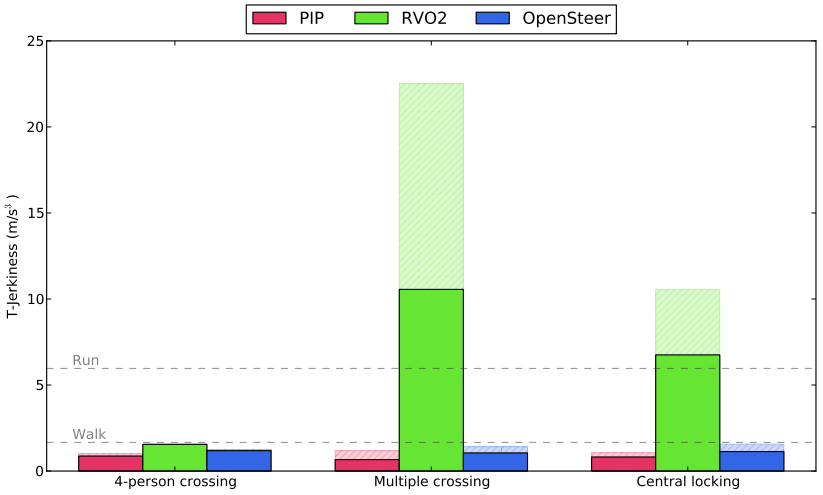
(b) Rate of change of tangential acceleration.

Figure 5.2: Normal (a) and tangential (b) jerk, the rate of the change of acceleration, measured in the "four-person crossing" experiment. The grey dashed lines are benchmark upper and lower limits measured from the "walking motion" database. Notice that while RVO2 (green) and OpenSteer (blue) cause high jerks beyond the upper and lower limits, PIP (red) remains within or close to the limits.

Figure 5.3 shows the normal and tangential jerkiness measured in the multi-character navigation experiment. Each block indicates the average case (solid) and worst case (halftone) acceleration jerkiness among the same group of individuals. I also measure the jerkiness in the "walking" and the "running" motion database, showing by the horizontal dashed lines. As seen in the figure, RVO2 generates motions that produce anomalously high jerkiness, particularly in the more complicated scenes such as "multiple crossing" and "central locking". The jerkiness of OpenSteer is reasonably low in the acceleration change of speed (t-jerkiness), but not in the change of direction (n-jerkiness). This also implies that if we were to generate locomotion animations along these paths using the "walking" motion database, it is unlikely motions can be fitted smoothly on their trajectories. By exploiting motions in the database, PIP can generate paths that produce a similar level of jerkiness in the database to guarantee the plausibility of the



(a) Normal (directional) jerkiness.



(b) Tangential (speed) jerkiness.

Figure 5.3: Benchmark results for change of behaviours. The grey dashed lines are the jerkiness (the standard deviation of jerk) measured in the "walking" and "running" motion database. RVO2 show very high jerkiness in both directional and speed changes in more crowded situations, indicating hyperactive and impulsive collision avoidance behaviours. The results of OpenSteer show that while it can maintain a steadier speed, it requires frequent change of direction in order to avoid collisions.

animation.

5.2.3 Proximity

To detect anomalous clearance between individuals, I define the *proximity* metric to find the nearest distance to the neighbours of each individual, and use the *minimum bounding radius* of the motion database as the benchmark to identify possible collisions. The minimum bounding radius is the radius

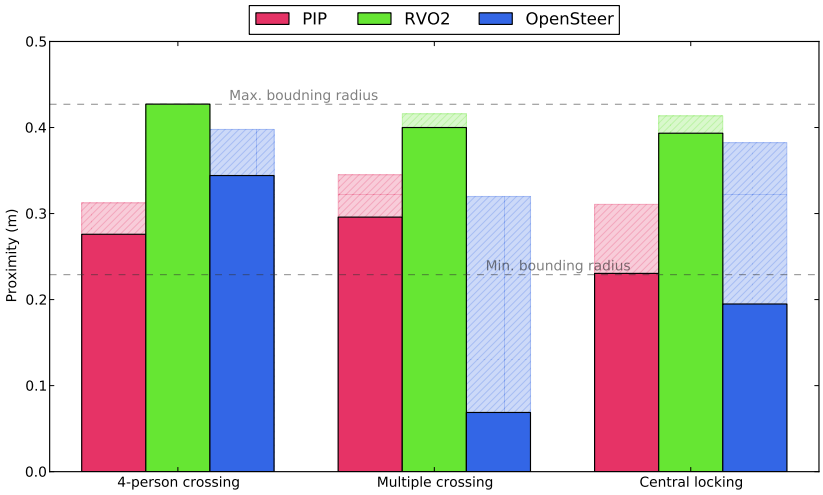


Figure 5.4: Benchmark results for interpersonal clearances. The grey dashed lines are maximum and minimum bounding radius measured in the "walking" motion database. OpenSteer shows severe inter-penetrations that violate the minimum bounding radius. RVO2 remains close to the maximum bounding radius, which could show excessive interpersonal spaces between individuals. The crowd of PIP can avoid each other closely without violating the minimum bounding radius.

of the minimum bounding circle found in the motion database, which is the near-least boundary an individual can occupy in the space using these motions. Violating this interpersonal clearance will almost certainly cause penetrations between individual and damage the visual plausibility of the animation. Given the nearest distance ℓ between a pair, we can consider two individuals are potentially collided with each other if ℓ is less than twice the radius of the minimum bounding circle.

Figure 5.4 shows the benchmark results for the proximity metric measured in multi-character navigation experiments. Each block indicates the minimal (solid) and the average (halftone) of the pair-wise nearest distances measured among each group. The distances are divided by two to be compared with the minimum and maximum bounding radius found in the walking database, showing by the horizontal dashed lines. Note that for RVO2 and OpenSteer, the maximum bounding radius was used as individual boundaries to avoid collisions. Therefore, in theory we should not expect any of their proximities lower than the maximum bounding radius. However, the results of both RVO2 and OpenSteer show undermined collision boundaries below the maximum bounding radius. Further, the graph indicate that there are potential collisions in "multiple crossing" and "central locking" experiments of OpenSteer. The interpersonal clearances of the results of PIP situate between the maximum and minimum bounding radius, with the inclination towards the minimum boundary. This indicates that the individuals can avoid each other very closely without causing collisions.

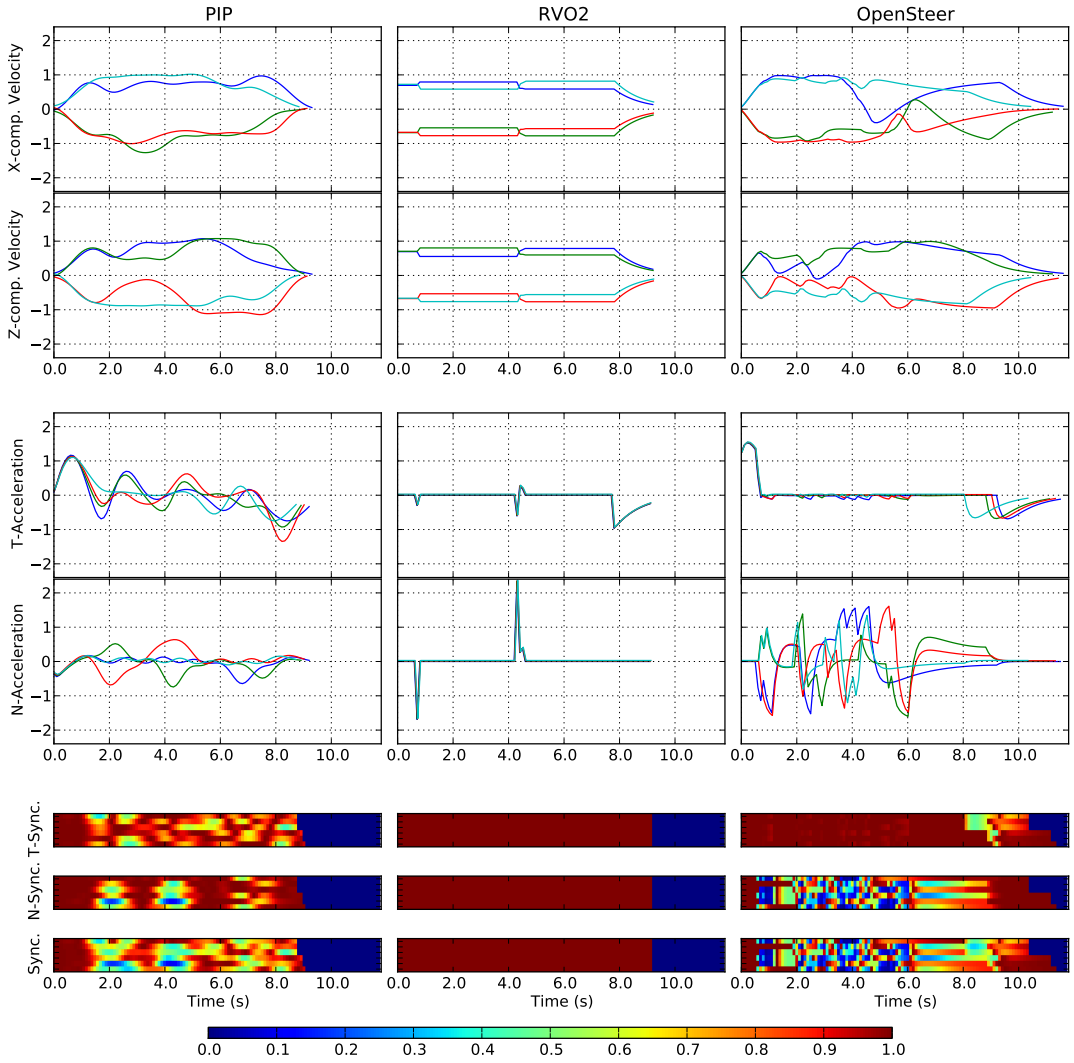


Figure 5.5: Pair synchronicities measured in four-person crossing: (top) The velocity profiles, (middle) the curvilinear acceleration profiles, and (bottom) the heat maps of pair synchronicity of the individuals, with dark red being highly synchronised movement. RVO2 shows identical synchronisation while OpenSteer shows little sign of coordination between pairs. PIP shows intermittent coordinations when pairs are avoiding each other.

5.2.4 Synchronicity

Finally, to identify anomalous hyper-synchronised group movements, I measure the "synchronicity" in crowd navigation animations. To achieve this, I measure the correlation of the acceleration between each pair of individuals to determine their pair synchronicity, and then analyse the distribution of the pair synchronicities among the crowd to estimate their overall synchronicity. For individual i and j, the pair synchronicity between the two is measured by

$$sync(t) = \frac{1}{1 + (a_i - a_j)^2}, \tag{5.3}$$

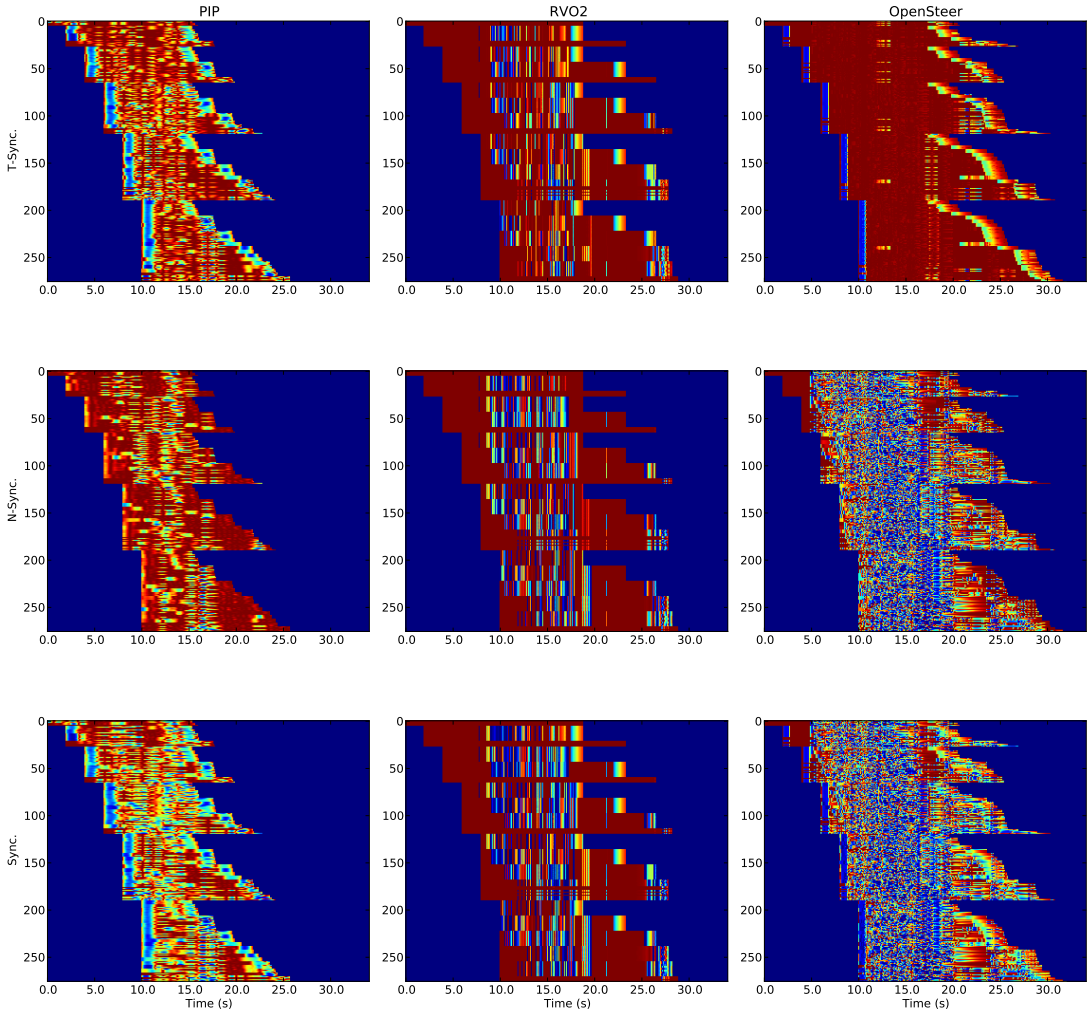


Figure 5.6: Pair synchronicities measured in multiple crossing. Pairs are arranged in the chronological order as each individual appears in the scene. RVO shows blocks of dark red regions indicating several long periods of identical synchronisation. The pairs in OpenSteer are highly synchronised in speed (t-sync) but not in direction (n-sync), resulting a disordered pair synchronicity with little coordination. The crowd of PIP shows patterns of occasional coordinations.

where a_i and a_j are their instantaneous accelerations at a given timestep t. Thus, two individuals are considered as momentarily synchronised when $sync(t)$ is close to one. Figure 5.5 shows the pair synchronicities and the corresponding velocities and accelerations measured in the "four-person crossing" experiment. Note that the synchronicities of the normal (n-sync.) and tangential (t-sync.) components are measured separately and are then multiplied to obtain their product so that two individuals are considered synchronised only when both of their directional and tangential accelerations are synchronised. While OpenSteer shows very little sign of synchronisation, the pair synchronicities in RVO2 are anomalously high, indicating identical movements. The results of PIP show intermittent synchronisations when

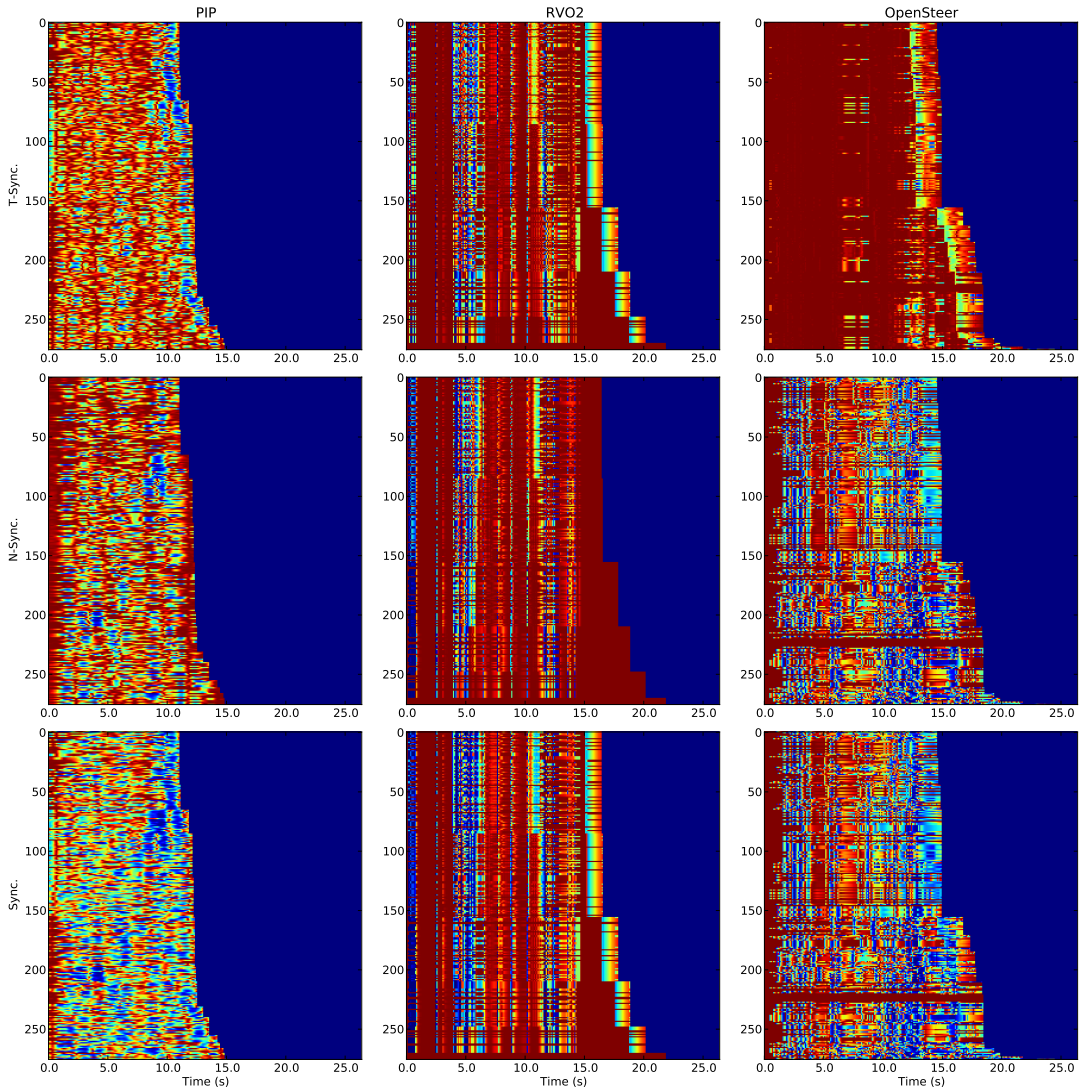
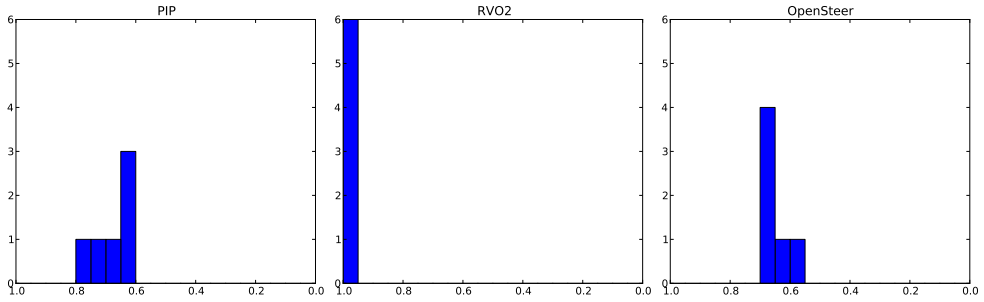


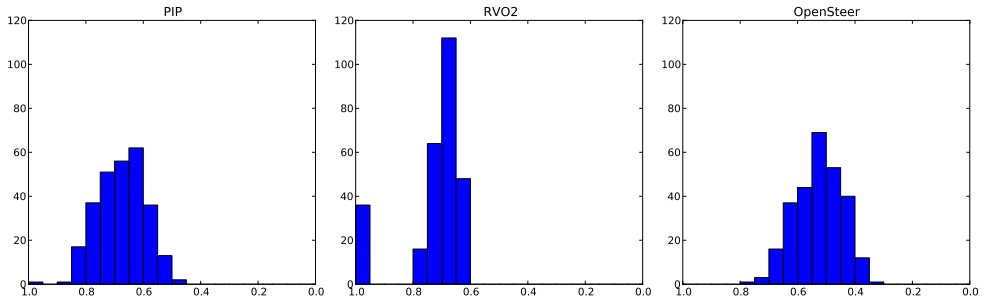
Figure 5.7: Pair synchronicities measured in central locking. The results suggest that overall the crowds are less synchronised/coordinated than crowds in multiple crossing as there are more individuals converge in the middle at the same time. RVO still produces periods of identical synchronised movements between many of its pairs.

individuals interact in close proximity. Figure 5.6 and Figure 5.7 show the pair synchronicities measured in the "multiple crossing"and "central locking" experiments.

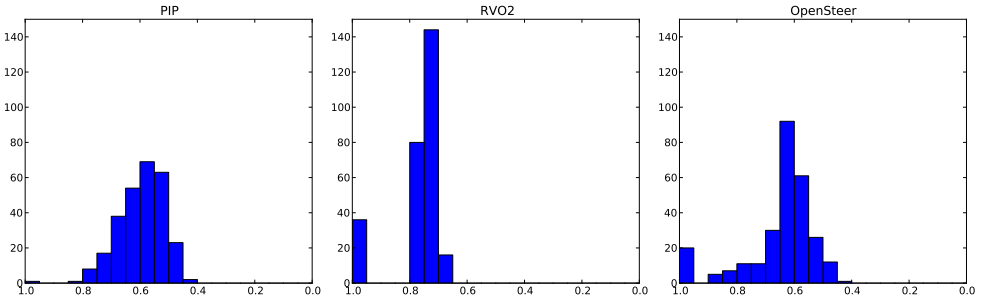
Now that I have computed the synchronicities for each pair, I can estimate the collective synchronicity among the crowd. This is done by analysing the pair-synchronicity distribution and estimating the cumulated density of the highly-synchronised pairs. I first calculate the expected value of the synchronicity for each pair by measuring its mean synchronicity. Each pair is now represented by a scalar $\bar{s}$ to indicate its averaged synchronicity which is between zero to one. I then find the distribution of $\bar{s}$ for all pairs by categorising them into n_{bin} bins between zero and one, and analysing the frequency of each bin. Figure 5.8 shows the histograms generated using the pair synchronicities measured in the multi-character nav-



(a) Four-person crossing.



(b) Multiple crossing.



(c) Central locking.

Figure 5.8: Histograms of pair-wise synchronicity. The distribution of pair synchronicity in PIP are situated between the range of 0.5 – 0.8, showing consistent levels of coordination. The data distribution varies noticeably with different crowd settings in OpenSteer. RVO2 produces more hyper-synchronised pairs ($\bar{s} > 0.95$) than PIP and OpenSteer.

igation experiments. The number of categories n_{bin} is chosen to be 20 so the interval between each bin is 0.5. While PIP and OpenSteer show more distributed densities across several different levels of pair synchronicities, RVO2 shows higher concentration on only few of the higher categories. More crucially, the results of RVO2 shows high frequency in the highest synchronised category (0.95 to 1.0), which indicates hyper-synchronised movements among a group of individuals.

The overall synchronicity is shown in Figure 5.9. Each represents the normalised cumulated density of the highly-synchronised pairs ($\bar{s} > 0.6$), with hyper-synchronised pairs ($\bar{s} > 0.95$) highlighted by the solid blocks atop. The results show that RVO2 produces motions with full synchronicities in all cases, with all pairs hyper-synchronised in “four-person crossing” and nearly 20% in “multiple crossing” and

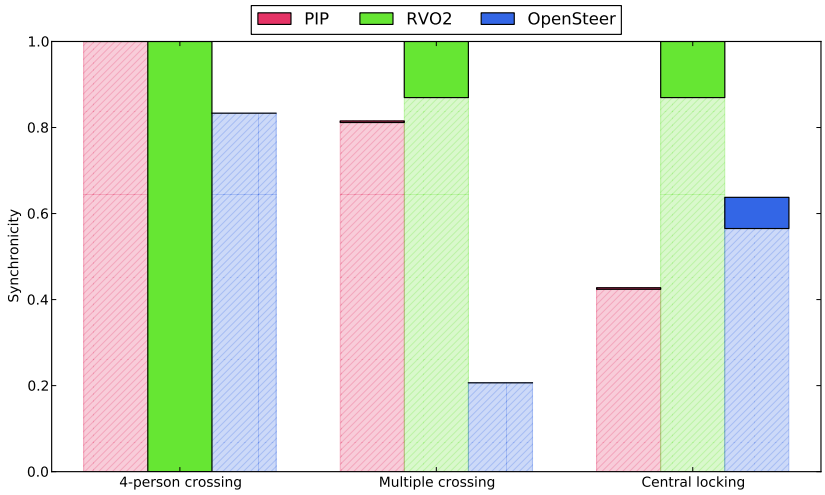


Figure 5.9: Benchmark results for group synchrony. Each bar indicates the overall synchronicity with the solid region highlighting the portion of the hyper-synchronised pairs. The synchronicity of PIP gradually decreases as the crowd becomes denser in the centre of the scene. RVO2 and OpenSteer both create hyper-synchronised groups.

"central locking". OpenSteer shows inconsistent results as the motions are relatively more dispersed in the first two experiments, but somehow generates more synchronised animations, including hyper-synchronised pairs, when the situation is more congested in the third experiment. PIP only show a small fraction of hyper-synchronised pairs. The results of PIP also demonstrates consistent synchronicity changes throughout the experiments in that it gradually deceases as the collision avoidance situation becomes more complicated.

5.3 Further Analysis

In this section, I apply the plausibility metrics to examine navigation animations generated using different navigation and collision-avoidance strategies. This includes the comparison of scene navigation techniques, and the analysis of the impacts on the multi-character navigation of various conditions. I revisit the navigation animations generated by path finding and following (PFF) and environment-aware motion graphs (EMG) in the "object-cluttered space" experiment in Chapter 3. To analyse the impact on the plausibility of crowd navigation, I focus on the "central locking" experiment using the PIP technique. I measure and compare the plausibilities between results generated by using various collision detection and congestion levels.

5.3.1 Scene Navigation Strategies

Figure 5.10 shows the resistance and jerkiness measurements of the navigation animations in Section 3.4.2. EMG demonstrates an overall lower resistance across ten experiments by using shorter paths,

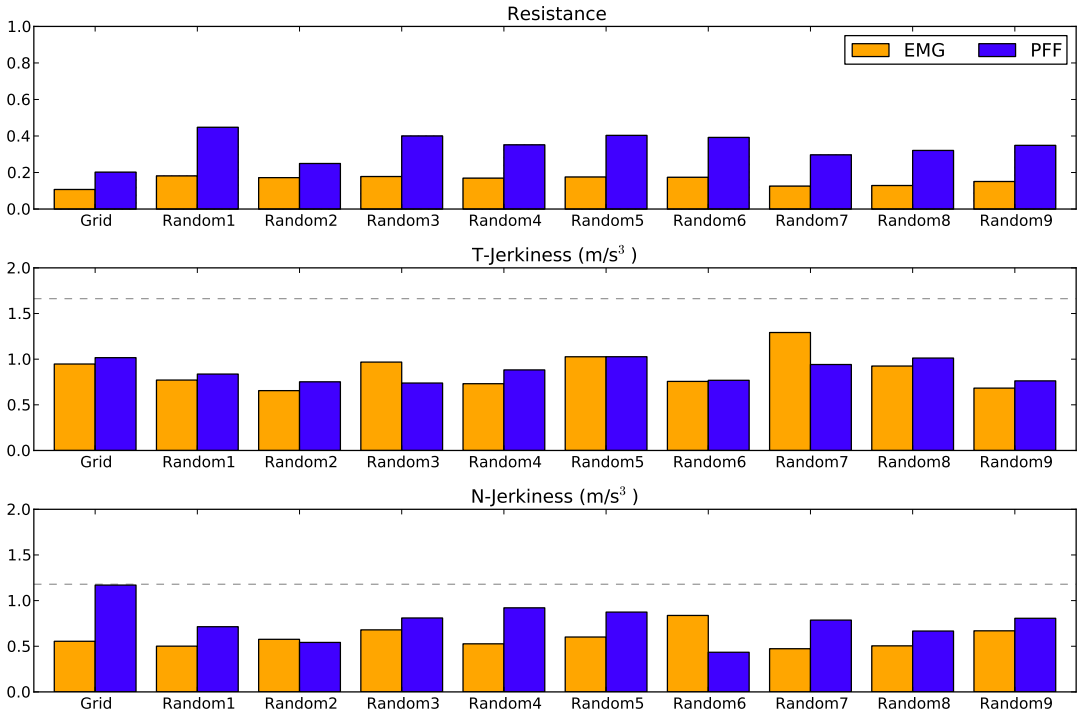


Figure 5.10: Benchmark comparison of "object-cluttered space" experiments in Chapter 3. The dashed lines are the benchmark jerkiness measured in the "walking" motion database. The resistance measurement shows that EMG requires less effort to avoid obstacles. While the measurements of EMG are mostly lower than PFF, both methods can produce low-jerkiness animations within the benchmarks.

while PFF requires higher energy to complete detoured paths to bypass same sets of obstacles. The jerkiness measured in PFF and EMG are both low and below the walking jerkiness benchmark (grey dashed lines). This is because all of their animations are generated using the motions from the "walking" motion database. However, the benchmarks show that overall, EMG produces lower jerkiness in most cases, despite the fact that it often chooses a path that involves more turning motions around obstacles.

5.3.2 Levels of Collision Detection

In this experiment, I measure the plausibilities between three levels of collision detection of the PIP method in the "central locking" experiment. I switch between maximum bounding circles (level 0), dynamic bounding circles (level 1), and full-body (level 2) collision-detection techniques to generate navigation animations, as shown in Figure 5.11. The results correctly indicate that the resistance and proximity measurements are inversely related to the levels of collision detections. The proximity of level 0 also indicates that each individual is potentially occupying a rather large personal space, leaving anomalously large clearances in-between. The synchronicity measures suggest that a more precise collision detection would increase the cooperation during interpersonal collision avoidance, since individuals are able to travel pass each other closely instead of constantly giving ways or making detours.

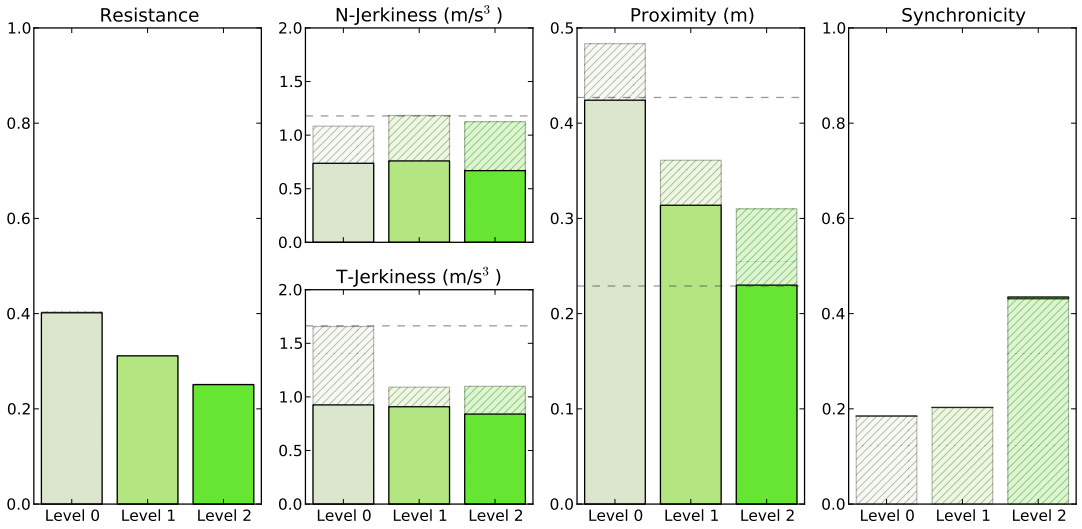


Figure 5.11: Benchmark comparison between three levels of collision detection: (level 0) maximum bounding circles, (level 1) dynamic bounding circles, and (level 2) full-body collision detection. Reducing the precision of collision detection results in higher resistance and jerkiness, and the decrease of the coordination between pairs.

5.3.3 Levels of Congestion

In the final experiment, I analyse the impact of the increase of the crowd density in the "central locking" situation. I increase the number of the individuals uniformly distributed around the starting circle from 4 individuals to 8, 16, and 24. Since each individual is asked to arrive at the opposite side of the circle, the congestion becomes more severe as the number of individuals increases. Figure 5.12 shows the results of the benchmark comparison. As seen in the figure, the resistance correctly shows positive correlation with the density of the crowd. Also, the synchronicity is identified as reversely related to the level of congested situations. In other words, the movements become relatively more random as the crowd size grows.

These results demonstrate that the plausibility metrics not only detect the anomalies in navigation animations, but also allow us to interpret their behaviours in a quantitative manner.

5.4 Discussion

In this chapter, I presented a quantitative approach to evaluate the plausibility of navigation animations. I devised a set of plausibility metrics to detect potential anomalous behaviours caused by collision avoidance. By measuring the curvilinear kinematics of individual movements as well as the spatio-temporal correlations between them, my metrics were able to compare between navigation animations generated using different collision detection models and collision avoidance strategies. While my approach evaluates the plausibility based on several benchmark measurements from the human motion database, it does

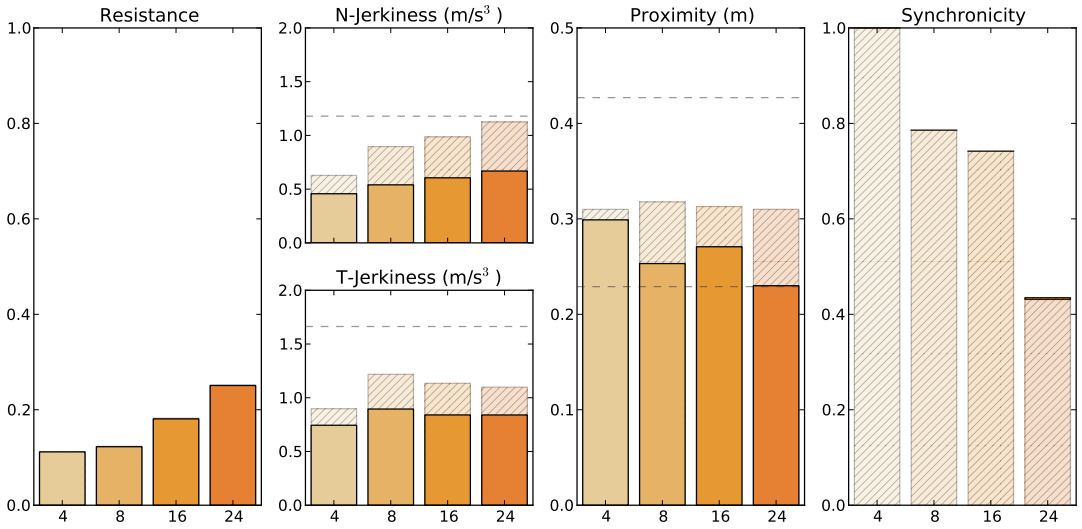


Figure 5.12: Benchmark comparison between different levels of congestion. The metrics correctly indicate that the effort to avoid collision increases as the scene becomes more crowded, so as the n-jerkiness due to more frequent changes of direction. Pair coordination also becomes rarer as shown in the synchronicity measure.

not rely on the resemblance to a particular type of navigation training data. Thus my crowd analysis is independent of the virtual environments or the settings of the crowd, and is hence more general.

My plausibility metrics evaluated navigation animations by comparing the biomechanical efforts, rate of change of behaviours, interpersonal clearances, and synchronised group movements between the crowds. The "resistance" measurement was able to estimate the excessive energies used to travel through obstacles. The normal and tangential "jerkiness" were measured to determine not only the frequency but also the amplitude of the acceleration variation of the directional and speed changes, respectively. To highlight potential collisions and excessive interpersonal spaces, I measured the "proximity" by finding the nearest distance between each pair in the crowd. I computed and compared pairwise "synchronicity" to measure pair synchronicities in the crowd in order to detect hyper-synchronous movements between crowd individuals.

My motion evaluation approach was an initial attempt to quantitatively analyse navigation and collision avoidance behaviours. However I believe there are other synthetic human motions that involve planning and collision avoidance could be evaluated in the same manner. Currently my plausibility metrics do not take into account individual preferences, such as the tendency of choosing a certain speed or avoiding dense regions. As a result they might be falsely highlighted as potential anomalies. To handle such exceptions will require further analysis. While these metrics were designed to evaluate computer-generated navigation animations, this approach could be extended to detect anomalous behaviours in real crowds. We could analyse the trajectories from the videos to highlight anomalous individual and group behaviours. This could be beneficial for security and surveillance applications. Another possible

extension is to establish certain crowd “profiles” by measuring real-world crowd movements with my metrics. For instance, we could compare different evacuation profiles to evaluate the safety design of a building. It would be very interesting to compare these profiles between different real-world crowd behaviours.

Chapter 6

Conclusion

In this thesis, I have demonstrated that the accuracy of collision avoidance plays a vital role in computer-generated human motions. To address this, I presented new techniques to optimise and evaluate the plausibility of navigation animations. I show that the *decoupling of motion planning and motion synthesis* and the *separation of interpersonal collision avoidance from path planning* are two major issues compromising the accuracy of collision avoidance, which in turn causes anomalous behaviours such as erratic travel paths, unnatural change of motions, and incorrect interpersonal clearances. My contribution in this thesis is therefore in describing new joint optimisation techniques that support precise collision detection and avoidance in character navigation and crowd interactions, as well as introducing a quantitative approach that evaluates the plausibility of navigation animations. In this final chapter, I summarise my results and discuss possible future directions of this research.

6.1 Summary

The aim of this thesis is to reduce anomalous behaviours in navigation and collision avoidance in character animation. My hypothesis is that by increasing the accuracy of collision avoidance, we will be able to minimise the anomalous behaviours, such as making excessive course and pace changes, in character navigation, and therefore the resulting animation will be more plausible. Here, I summarise the contribution of the new techniques presented in this thesis and discuss how they support the hypothesis.

My environment-aware motion graph (EMG) technique achieves precise collision avoidance by directly searching for motions that avoids obstacles in the motion capture database. It does so by jointly optimising the sequence of motions towards being goal-reaching and collision-free. It is through this joint optimisation that full-body collision detection and near-optimal collision avoidance are realised in motion database queries. The result is a combined motion-planning and motion-synthesis stage. In doing so my technique prevents anomalous collision-avoidance behaviours from choosing a sub-optimal path using a simplified collision model and approximating this path with ill-fitting motions in the conventional decoupled approach.

Fundamentally, the underlying database query is a randomised motion graph search, that globally

looks for solutions that minimise the cost function. I incorporate collision errors in the joint cost function, which supports multiple static obstacles and the trajectories of the crowd movements. As a result, my method is able to plan global navigation in complex and dynamic environments.

My path interference perturbation (PIP) technique unifies goal navigation and interpersonal collision avoidance. I view crowd navigation as per-path global planning, where each path is an individual motion query that planned against the known trajectories of the crowd movements. Hence the crowd members anticipate collisions correctly and avoid each other closely, and individual animation is plausible without unnatural motion changes. I solve crowd interdependency by incrementally isolating groups where paths interfere as spatio-temporal constraints and iteratively re-planning their group interactions. By minimising the local and individual efforts to avoid collisions, my approach is able to maximise regional traffic flows while maintains the overall flow structure. Thus, my results are more exempt from the anomalous behaviours such as dispersed formations and frequent congestions caused by traditional local navigation models.

Finally, I introduce a quantitative approach to evaluate the plausibility of navigation animations. My plausibility metrics benchmark navigation animations by highlighting potential anomalies in individual and collective behaviours. I measure quantitative properties such as biomechanical energies and the rate of change of accelerations of the trajectories as well as their interpersonal clearances and pair synchroncities to identify anomalous behaviours caused by choosing sub-optimal motions to avoid collisions. The capability of evaluating the interaction between individuals in a quantitative manner enables us to interpret crowd behaviours and to identify motion anomalies more accurately. Unlike previous methods, I consider overall navigation behaviours rather than the reaction to single collision events or the resemblance to a particular type of crowd configuration. For collective behaviours, I measure pairwise correlation of the synchronous movements to highlight hyper-synchronised group motions. My observation is that for individuals to avoid each other, a certain level of coordination is required. Therefore I believe it is more indicative to determine how the individuals are synchronised together rather than to maximise the randomness in order to break the uniform pattern of the crowd. My plausibility metrics is general in that the measurements only rely on the analysis of the curvilinear motions of the trajectories, and as a result they are independent of the complexity of the environment or the size of the crowd.

6.2 Future Work

There are many interesting avenues for future research on the joint optimisation approach discussed in this thesis. One immediate extension is to consider planning more complex behaviours by incorporating a larger motion repertoire in the database. Currently, the motion query searches the entire graph for optimal solutions. We can select and switch between several sub-graphs for different terrains or crowded situation. An obvious example would be to drop sub-graphs that comprises running motions when the environment is cluttered with obstacles or is crowded with people. This is challenging as it would require high-level spatio-temporal reasoning to analyse the surroundings in order to decompose motion queries.

More types of interpersonal interactions other than collision avoidance can also be considered to extend this approach. Currently, my planning framework allows individuals avoid each other using different motion graphs. This is also achieved via the joint optimisation of the cumulative individual costs and the collision errors due to their path interferences. We can generalise the latter term as group constraints to include other types of interactions, such as leader following or pursuit-evade adversary actions, and the optimisation will attempt to find a solution that minimises individual efforts. From the results presented in this thesis, I am convinced that my approach can still generate more plausible animations than local collision-avoidance models when it is extended to support other types of interactions.

One fundamental challenge is to plan navigation for larger crowds. The main emphasis of this thesis is to reduce anomalies in navigation animations, and indeed my approach is centred on increasing the precision of human-environment and human-human interactions to improve the plausibility of individual and collective behaviours. To scale my approach to support planning for larger crowds would require whether or not to sacrifice precision, depending on the scale and the application, without compromising the plausibility. For a medium-sized (10-100) crowd in animation choreography, we can break down the crowd into a hierarchical composition, where primary members such as hero characters are planned first using full-body collision avoidance to interact with each other, while secondary members would then avoid the trajectories of the primary members and interact with others using a lower level-of-detail collision avoidance such dynamic bounding circles. For more than a hundred individuals, full-body collision detection would become inapplicable. Fortunately, for crowds to be observed at this scale, only severe collisions such as the inter-penetration of the human-body torsos are noticeable. From Chapter 5 we learnt that the interpersonal clearances of the results using the full-body collision model (The PIP method) tend to be inclined towards the minimum bounding circle found in the motion database. This gives us an insight to trade off collision precision for computational complexity by switching the collision model to the inner rigid bounding circle covering the torso to plan close interactions. This reduced collision model raises new research questions: can we obtain results with similar plausibility with this model instead of full-body collision detection? How severe are these supposedly limb collisions introduced by this model and how are they perceived visually at various scale? To be able to answer these questions would determine the scalability of my crowd planning approach.

My plausibility metrics can be applied to a broader range of crowd analysis. One direct application is to analyse real-world crowd motions in surveillance studies. As well as identifying individual and collective anomalies, we could also analyse the composition of the crowd by looking at the spectrum of the distribution of the metrics to determine any signs of emergent behaviours that are not necessarily spatio-temporal related. Pedestrian planning is another area where my metrics might be beneficial. For example, we could test different floor plans by evaluating the energy and change of behaviours of the crowd movements to refine the design. Another interesting future work is to consider these metrics as higher-level crowd controls. My plausibility metrics evaluate anomalous behaviours caused by sub-optimal collision avoidance motions. We can incorporate these metrics in the cost function to control overall crowd movements.

I formulate crowd navigation as a centralised planning of global paths for multiple moving entities. Unlike previous centralised methods, my approach does not rely on prioritising individuals or partitioning sub-groups in order to plan global paths. As a result, my method is able to find global solutions in situations where the paths of individuals do not have an obvious formation or structure. Therefore, I believe this method provides an alternative approach to general global planning problems such as multi-agent planning in robotics and artificial intelligence. These problems used to rely on decentralised methods that negotiate among agents for a plan to complete certain tasks, but using distributed models can show instability and inefficiency when the conflict between individual goals becomes severe. My planning framework prioritises local interactions in such way that when it re-plans these interactions, it minimises the disturbance to the system. The joint optimisation minimises the effort of local coordination as well as individual costs. I am convinced that these strategies can be extended to solve a much wider range of global planning problems.

Appendix A

Supplementary Videos

The supplementary videos are recorded in an optical disc of the DVD format. An alternative on-line version is available at: `www.cs.ucl.ac.uk/staff/J.Chen/thesis.html`. The contents of the supplementary videos are as follows:

A.1 Environment-aware Motion Graphs

A.2 Path Interference Perturbation

Bibliography

Arikan, O. (2006), Compression of motion capture databases, *ACM Trans. Graph.* **25**, 890–897.

Arikan, O. and Forsyth, D. A. (2002), Interactive motion generation from examples, *ACM Trans. Graph.* **21**, 483–490.

Bandi, S. and Thalmann, D. (1998), Space Discretization for Efficient Human Navigation.

Barraquand, J., Langlois, B. and Latombe, J.-C. (1991), Numerical potential field techniques for robot path planning, *Advanced Robotics, 1991. 'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on* , pp.1012 –1017 vol.2.

Beaudoin, P., Coros, S., van de Panne, M. and Poulin, P. (2008), Motion-motif graphs, *in Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.117–126.

Boulic, R., Magnenat-Thalmann, N. and Thalmann, D. (1990), A global human walking model with real-time kinematic personification, *Vis. Comput.* **6**(6), 344–358.

Bowden, R. (2000), Learning Statistical Models of Human Motion.

Brand, M. and Hertzmann, A. (2000), Style machines, *in SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp.183–192.

Bruderlin, A. and Calvert, T. W. (1989), Goal-directed, dynamic animation of human walking, *SIGGRAPH Comput. Graph.* **23**(3), 233–242.

Chen, J.-R. and Steed, A. (2011), Planning plausible human animation with environment-aware motion sampling, *in Proceedings of the 4th international conference on Motion in Games*, MIG'11, Springer-Verlag, Berlin, Heidelberg, pp.51–62.

Chen, J.-R. and Steed, A. (2014), Joint motion planning and motion synthesis for multi-character navigation (under submission).

Chenney, S. and Forsyth, D. A. (2000), Sampling plausible solutions to multi-body constraint problems, *in Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp.219–228.

Choi, M. G., Lee, J. and Shin, S. Y. (2003), Planning biped locomotion using motion capture data and probabilistic roadmaps, *ACM Trans. Graph.* **22**(2), 182–203.

Choset, H. and Burdick, J. (2000), Sensor-Based Exploration: The Hierarchical Generalized Voronoi Graph.

CMU Graphics Lab (2009), Carnegie Mellon University Motion Capture Database. [Online; announced April 15 2009].

Coros, S., Beaudoin, P. and van de Panne, M. (2009), Robust task-based control policies for physics-based characters, *in ACM SIGGRAPH Asia 2009 papers*, SIGGRAPH Asia '09, ACM, New York, NY, USA, pp.170:1–170:9.

Courty, N. and Corpetti, T. (2007), Crowd motion capture, *Comput. Animat. Virtual Worlds* **18**(4-5), 361–370.

da Silva, M., Abe, Y. and Popovic, J. (2008), Simulation of Human Motion Data using Short-Horizon Model-Predictive Control, *Comput. Graph. Forum* **27**(2), 371–380.

Ennis, C., Peters, C. and O'Sullivan, C. (2008), Perceptual evaluation of position and orientation context rules for pedestrian formations, *in Proceedings of the 5th symposium on Applied perception in graphics and visualization*, APGV '08, ACM, New York, NY, USA, pp.75–82.

Epic Games Inc. (2009), Unreal Engine, version 3. [Online; annouced November 18, 2008].

Geraerts, R. and Overmars, M. H. (2002), A comparative study of probabilistic roadmap planners, *in IN: WORKSHOP ON THE ALGORITHMIC FOUNDATIONS OF ROBOTICS*, pp.43–57.

Goldenstein, S., Karavelas, M., Metaxas, D., Guibas, L., Aaron, E. and Goswami, A. (2001), Scalable Nonlinear Dynamical Systems for Agent Steering and Crowd Simulation.

Gran, C. A., Vzquez, P.-P. and Gonzlez, M. F. (2004), Way-Finder: Guided Tours Through Complex Walkthrough Models., *Comput. Graph. Forum* **23**(3), 499–508.

Guy, S. J., Chhugani, J., Curtis, S., Dubey, P., Lin, M. and Manocha, D. (2010), PLEdestrians: a least-effort approach to crowd simulation, *in Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.119–128.

Guy, S. J., Chhugani, J., Kim, C., Satish, N., Lin, M., Manocha, D. and Dubey, P. (2009), ClearPath: highly parallel collision avoidance for multi-agent simulation, *in Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, ACM, New York, NY, USA, pp.177–187.

Guy, S. J., van den Berg, J., Liu, W., Lau, R., Lin, M. C. and Manocha, D. (2012), A statistical similarity measure for aggregate crowd dynamics, *ACM Trans. Graph.* **31**(6), 190:1–190:11.

Havok Inc. (2008), Havok Behavior animation system, version 5.5. [Online; annouced November 18, 2008].

Helbing, D. and Molnar, P. (1995), Social Force Model for Pedestrian Dynamics, *PHYSICAL REVIEW E* **51**, 4282.

Henry, J., Shum, H. P. H. and Komura, T. (2012), Environment-aware real-time crowd control, *in Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.193–200.

Hodgins, J. K., Wooten, W. L., Brogan, D. C. and O'Brien, J. F. (1995), Animating human athletics, *in Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '95, ACM, New York, NY, USA, pp.71–78.

Ikemoto, L., Arikan, O. and Forsyth, D. (2006), Knowing when to put your foot down, *in Proceedings of the 2006 symposium on Interactive 3D graphics and games*, I3D '06, ACM, New York, NY, USA, pp.49–53.

Ikemoto, L., Arikan, O. and Forsyth, D. (2007), Quick transitions with cached multi-way blends, *in Proceedings of the 2007 symposium on Interactive 3D graphics and games*, I3D '07, ACM, New York, NY, USA, pp.145–151.

Ikemoto, L. and Forsyth, D. A. (2004), Enriching a motion collection by transplanting limbs, *in Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '04, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.99–108.

Ju, E., Choi, M. G., Park, M., Lee, J., Lee, K. H. and Takahashi, S. (2010), Morphable crowds, *ACM Trans. Graph.* **29**(6), 140:1–140:10.

Kalisiak, M. and van de Panne, M. (2001), A Grasp-based Motion Planning Algorithm for Character Animation.

Kallmann, M. (2010), Shortest paths with arbitrary clearance from navigation meshes, *in Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.159–168.

Kamphuis, A. and Overmars, M. H. (2004), Finding paths for coherent groups using clearance, *in Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '04, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.19–28.

Kavraki, L. E., Svestka, P., Vestka, L. E. K. P., claude Latombe, J. and Overmars, M. H. (1996), Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces, *IEEE Transactions on Robotics and Automation* **12**, 566–580.

Kavraki, L., Kolountzakis, M. N. and Latombe, J.-C. (1998), Analysis of Probabilistic Roadmaps for Path Planning.

Kim, M., Hwang, Y., Hyun, K. and Lee, J. (2012), Tiling motion patches, *in Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.117–126.

Kim, M., Hyun, K., Kim, J. and Lee, J. (2009), Synchronized multi-character motion editing, *ACM Trans. Graph.* **28**(3), 79:1–79:9.

Ko, H. and Badler, N. I. (1996), Animating Human Locomotion with Inverse Dynamics, *IEEE Comput. Graph. Appl.* **16**(2), 50–59.

Koga, Y., Kondo, K., Kuffner, J. and Latombe, J.-C. (1994), Planning motions with intentions, *in Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, SIGGRAPH '94, ACM, New York, NY, USA, pp.395–408.

Kovar, L., Gleicher, M. and Pighin, F. (2002a), Motion graphs, *in SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, pp.473–482.

Kovar, L., Schreiner, J. and Gleicher, M. (2002b), Footskate cleanup for motion capture editing, *in Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '02, ACM, New York, NY, USA, pp.97–104.

Kuffner, Jr., J. J. (1998), Goal-Directed Navigation for Animated Characters Using Real-Time Path Planning and Control, *in Proceedings of the International Workshop on Modelling and Motion Capture Techniques for Virtual Environments*, CAPTECH '98, Springer-Verlag, London, UK, UK, pp.171–186.

Kwon, T., Lee, K. H., Lee, J. and Takahashi, S. (2008), Group motion editing, *ACM Trans. Graph.* **27**(3), 80:1–80:8.

Laszlo, J., van de Panne, M. and Fiume, E. (1996), Limit cycle control and its application to the animation of balancing and walking, *in Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, ACM, New York, NY, USA, pp.155–162.

Lau, M. and Kuffner, J. J. (2005), Behavior planning for character animation, *in Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '05, ACM, New York, NY, USA, pp.271–280.

Lau, M. and Kuffner, J. J. (2006), Precomputed search trees: planning for interactive goal-driven animation, *in Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '06, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.299–308.

Lee, J., Chai, J., Reitsma, P. S. A., Hodgins, J. K. and Pollard, N. S. (2002), Interactive control of avatars animated with human motion data, *ACM Trans. Graph.* **21**(3), 491–500.

Lee, J. and Lee, K. H. (2004), Precomputing avatar behavior from human motion data, *in Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '04, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.79–87.

Lee, K. H., Choi, M. G. and Lee, J. (2006), Motion patches: building blocks for virtual environments annotated with motion data, *in ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, ACM, New York, NY, USA, pp.898–906.

Lee, K. H., Choi, M. G., Hong, Q. and Lee, J. (2007), Group behavior from video: a data-driven approach to crowd simulation, *in Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '07, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.109–118.

Lee, Y., Lee, S. J. and Popović, Z. (2009), Compact character controllers, *ACM Trans. Graph.* **28**(5), 169:1–169:8.

Liu, C. K., Hertzmann, A. and Popović, Z. (2006), Composition of complex optimal multi-character motions, *in Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '06, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.215–222.

Lo, W.-Y. and Zwicker, M. (2008), Real-time planning for parameterized human motion, *in Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.29–38.

Lo, W.-Y. and Zwicker, M. (2010), Bidirectional Search for Interactive Motion Synthesis, *Computer Graphics Forum* **29**(2).

Lo, W.-Y., Knaus, C. and Zwicker, M. (2012), Learning motion controllers with adaptive depth perception, *in Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.145–154.

Lozano-Pérez, T. and Wesley, M. A. (1979), An algorithm for planning collision-free paths among polyhedral obstacles, *Commun. ACM* **22**(10), 560–570.

Massive Software (2009), Massive (Multiple Agent Simulation System in Virtual Environment), version 3.5. [Online; 2009].

McDonnell, R., Larkin, M., Dobbyn, S., Collins, S. and O'Sullivan, C. (2008), Clone attack! Perception of crowd variety, *in ACM SIGGRAPH 2008 papers*, SIGGRAPH '08, ACM, New York, NY, USA, pp.26:1–26:8.

Mizuguchi, M., Buchanan, J. and Calvert, T. (2001), Data driven motion transitions for interactive games.

Mordatch, I., de Lasa, M. and Hertzmann, A. (2010), Robust physics-based locomotion using low-dimensional planning, *ACM Trans. Graph.* **29**(4), 71:1–71:8.

Morini, F., Yersin, B., Maym, J. and Thalmann, D. (2007), Real-Time Scalable Motion Planning for Crowds, *in Proceedings of the 2007 International Conference on Cyberworlds*, CW '07, IEEE Computer Society, Washington, DC, USA, pp.144–151.

Muico, U., Lee, Y., Popović, J. and Popović, Z. (2009), Contact-aware nonlinear control of dynamic characters, *ACM Trans. Graph.* **28**(3), 81:1–81:9.

Narain, R., Golas, A., Curtis, S. and Lin, M. C. (2009), Aggregate dynamics for dense crowd simulation, *ACM Trans. Graph.* **28**(5), 122:1–122:8.

NaturalMotion Ltd. (2009), Morpheme animation middleware, version 2. [Online; announced April 15 2009].

Nieuwenhuisen, D., Kamphuis, A. and Overmars, M. H. (2007), High quality navigation in computer games, *Sci. Comput. Program.* **67**(1), 91–104.

Ondřej, J., Pettré, J., Olivier, A.-H. and Donikian, S. (2010), A synthetic-vision based steering approach for crowd simulation, *ACM Trans. Graph.* **29**(4), 123:1–123:9.

Oshita, M. and Ogiwara, Y. (2009), Sketch-Based Interface for Crowd Animation, *in Proceedings of the 10th International Symposium on Smart Graphics*, SG '09, Springer-Verlag, Berlin, Heidelberg, pp.253–262.

Pettré, J., Laumond, J.-P. and Siméon, T. (2003), A 2-stages locomotion planner for digital actors, *in Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '03, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.258–264.

Pettré, J., Laumond, J.-P. and Thalmann, D. (2005), A navigation graph for real-time crowd animation on multilayered and uneven terrain., *in Proc. The First International Workshop on Crowd Simulation (V-CROWDS'05)*.

Pettré, J., Ondřej, J., Olivier, A.-H., Cretual, A. and Donikian, S. (2009), Experiment-based modeling, simulation and validation of interactions between virtual walkers, *in Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, ACM, New York, NY, USA, pp.189–198.

Pettre, J., Simeon, T. and Laumond, J.-P. (2002), Planning human walk in virtual environments, *in Proc. IEEE/RSJ International Conference on Intelligent Robots and System (IROS)*, Vol. 3, pp.3048–3053.

Pražák, M. and O'Sullivan, C. (2011), Perceiving human motion variety, *in Proceedings of the ACM SIGGRAPH Symposium on Applied Perception in Graphics and Visualization*, APGV '11, ACM, New York, NY, USA, pp.87–92.

Pullen, K. and Bregler, C. (2002), Motion capture assisted animation: texturing and synthesis, *ACM Trans. Graph.* **21**(3), 501–508.

Raibert, M. H. and Hodgins, J. K. (1991), Animation of dynamic legged locomotion, *SIGGRAPH Comput. Graph.* **25**(4), 349–358.

Reitsma, P. S. A. and Pollard, N. S. (2003), Perceptual metrics for character animation: sensitivity to errors in ballistic motion, *in ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, ACM, New York, NY, USA, pp.537–542.

Reitsma, P. S. A. and Pollard, N. S. (2004), Evaluating motion graphs for character navigation, *in Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '04, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.89–98.

Ren, C., Zhao, L. and Safonova, A. (2010), Human Motion Synthesis with Optimization-based Graphs., *Comput. Graph. Forum* **29**(2), 545–554.

Ren, L., Patrick, A., Efros, A. A., Hodgins, J. K. and Rehg, J. M. (2005), A data-driven approach to quantifying natural human motion, *ACM Trans. Graph.* **24**, 1090–1097.

Reynolds, C. (1999), Steering Behaviors For Autonomous Characters.

Reynolds, C. W. (1987), Flocks, herds and schools: A distributed behavioral model, *SIGGRAPH Comput. Graph.* **21**(4), 25–34.

Safonova, A. and Hodgins, J. K. (2005), Analyzing the physical correctness of interpolated human motion, *in Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '05, ACM, New York, NY, USA, pp.171–180.

Safonova, A. and Hodgins, J. K. (2007), Construction and optimal search of interpolated motion graphs, *ACM Trans. Graph.* **26**(3).

Schödl, A., Szeliski, R., Salesin, D. H. and Essa, I. (2000), Video textures, *in SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp.489–498.

Shao, W. and Terzopoulos, D. (2005), Autonomous pedestrians, *in Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '05, ACM, New York, NY, USA, pp.19–28.

Shum, H. P. H., Komura, T., Shiraishi, M. and Yamazaki, S. (2008), Interaction patches for multicharacter animation, *in ACM SIGGRAPH Asia 2008 papers*, SIGGRAPH Asia '08, ACM, New York, NY, USA, pp.114:1–114:8.

Snape, J., Guy, S. J. and van den Berg, J. (2010), Independent navigation of multiple robots and virtual agents, *in Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1 - Volume 1*, AAMAS '10, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp.1645–1646.

Tanco, L. M. and Hilton, A. (2000), Realistic synthesis of novel human movements from a database of motion capture examples, *in HUMO '00: Proceedings of the Workshop on Human Motion (HUMO'00)*, IEEE Computer Society, Washington, DC, USA, p.137.

Treuille, A., Cooper, S. and Popović, Z. (2006), Continuum crowds, *ACM Trans. Graph.* **25**(3), 1160–1168.

Treuille, A., Lee, Y. and Popović, Z. (2007), Near-optimal character animation with continuous control, *ACM Trans. Graph.* **26**(3).

van den Berg, J., Lin, M. C. and Manocha, D. (2008a), Reciprocal Velocity Obstacles for Real-Time Multi-Agent Navigation, *in IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION*, IEEE, pp.1928–1935.

van den Berg, J., Patil, S., Sewall, J., Manocha, D. and Lin, M. (2008b), Interactive navigation of multiple agents in crowded environments, *in Proceedings of the 2008 symposium on Interactive 3D graphics and games*, I3D '08, ACM, New York, NY, USA, pp.139–147.

van Toll, W. G., Cook, IV, A. F. and Geraerts, R. (2012), Real-time density-based crowd simulation, *Comput. Animat. Virtual Worlds* **23**(1), 59–69.

Wang, J. and Bodenheimer, B. (2003), An evaluation of a cost metric for selecting transitions between motion segments, *in Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '03, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.232–238.

Wang, J. and Bodenheimer, B. (2004), Computing the duration of motion transitions: an empirical approach, *in Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '04, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.335–344.

Yersin, B., Maïm, J., Pettré, J. and Thalmann, D. (2009), Crowd patches: populating large-scale virtual environments for real-time applications, *in Proceedings of the 2009 symposium on Interactive 3D graphics and games*, I3D '09, ACM, New York, NY, USA, pp.207–214.

Yin, K., Loken, K. and van de Panne, M. (2007), SIMBICON: simple biped locomotion control, *ACM Trans. Graph.* **26**.

Zhao, L. and Safonova, A. (2008), Achieving good connectivity in motion graphs, *in Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp.127–136.

Zhao, L., Normoyle, A., Khanna, S. and Safonova, A. (2009), Automatic construction of a minimum size motion graph, *in Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, ACM, New York, NY, USA, pp.27–35.