Run and Catch: Dynamic Object-Catching of Quadrupedal Robots

Yangwei You[1], Tianlin Liu[1*], Xiaowei Liang[1], Zhe Xu[1], Mingliang Zhou[1], Zhibin Li[2], Shiwu Zhang[3]

***Abstract*— Quadrupedal robots are performing increasingly more real-world capabilities, but are primarily limited to locomotion tasks. To expand their task-level abilities of object acquisition, i.e., run-to-catch as frisbee catching for dogs, this paper developed a control pipeline using stereo vision for legged robots which allows for dynamic catching balls while the robot is in motion. To achieve high-frame-rate tracking, we designed a ball that can actively emit homogeneous infrared (IR) light and then located the flying ball based on binocular vision positioning using the onboard RealSense D450 camera with an additional IR bandpass filter. The camera was mounted on top of a 2-DoF head to gain a full view of the target ball. A state estimation module was developed to fuse the vision positioning, camera motor readings, localization result of RealSense T265 equipped on the back, and the legged odometry output altogether. With the use of a ballistic model, we achieved a robust estimation of both the ball and robot positions in an inertial coordinate. Additionally, we developed a close-loop catching strategy and employed trajectory prediction so that tracking and run-to-catch were performed simultaneously, which is critical for such drastically dynamic and precise tasks. The proposed approach was validated through both static testing and dynamic catch experiments conducted on the *CyberDog* robot with a high success rate.**

I. Introduction

Object catching is a significant challenge for legged robots, as it requires precise and real-time state estimation, coupled with the generation of highly dynamic motions and maintaining balance using legs. Although there has been a long history of study on this task [1], [2], [3], [4], [5], only a handful of works have focused on legged robots. Compared to fixed-based manipulators or wheeled mobile platforms, the nature of discrete foot-ground contacts from legged locomotion has a much higher level of noise in odometry, coupled with complex floating-base dynamics. In this context, our work investigated ball-catching tasks for a quadruped robot, and developed a full stack solution including a fast vision system and a novel state estimation to simultaneously localize both the robot and flying ball, which demonstrated high success rates for ball-catching tasks using only onboard sensors.

A. Related work

For fixed-base systems, ball catching was accomplished with a 4 degree of freedom (DoF) manipulator [6], [7]. The desired catching point is obtained by combining parabola curve prediction with error correction. Another ball catcher system is developed by Frese et al., who proposed an Extended Kalman filter (EKF) to track and predict the target catching position [8]. One of the state-of-the-art works is achieved by an EKF estimator that localizes ball position [9]. Another catching research is done by a fix-based humanoid robot, where a linear Kalman filter is re-initialized every time a new ball is thrown and predicts the ball destination and catch timing [10]. Though all those works achieved impressive results, the noise and uncertainties of the localization of fixed-based robots are much less challenging compared to legged robots in motion.

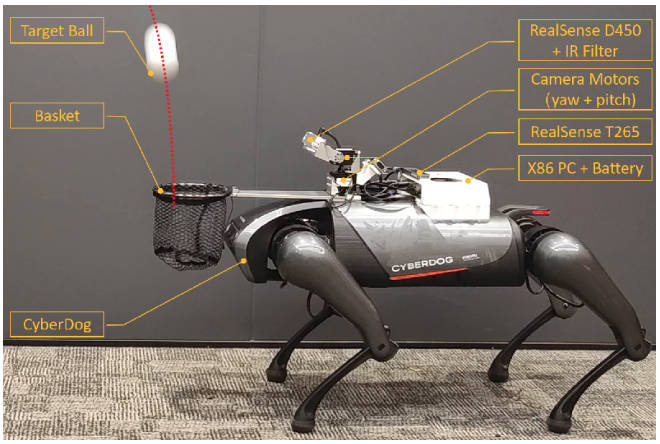


Fig. 1: System overview of *CyberDog* for ball-catching. A RealSense D450 camera with IR bandpass filters is mounted on a 2-DoF head and the basket is at its front. One x86 computer and its battery are placed on the back with a RealSense T265 tracking module on top. Red dotted line is the target ball's trajectory.

Due to its simplicity, the quadrotor has been widely researched for ball-catching. Muller et al. presented a Kalman filter based estimation to predict the state of balls [11]. With a proposed learning-based model predictive controller and EKF, Bouffard et al. demonstrated the possibility of catching a ball with a quadrotor [12]. However, perception with a ground-based vision system needs careful calibration before the task, which imposes additional limitations. With only an onboard camera, a reinforcement learning (RL) based approach is applied by Silva et al. which estimates the ball position and velocity by camera feed and determines the hitting point [13]. Another research uses an unscented Kalman filter (UKF) based estimator to fuse optical flow and inertial data by a simple 2-D visual ball tracker [14]. But due to the differences in dynamics, control strategies, and sensors, these algorithms from quadrotors cannot be directly transferred to legged locomotion.

A mobile ball catcher was developed by building an

[1] Xiaomi Robotics Lab, Xiaomi Sci-Tech Park, Anningzhuang Road, Haidian District, Beijing, 100085, P.R.China.
[2] University College London, Department of Computer Science, London WC1E 6BT, UK
[3] University of Science and Technology of China, No.96, JinZhai Road Baohe District, Hefei, Anhui, 230026, P.R.China.
* Corresponding author. Email:liutianlin1@xiaomi.com.

active stereo camera to track flying balls [15], which was recently improved by applying deep learning to initialize the Kalman filter estimator [16]. However, the camera is on the ground, not onboard, which does not have uncertainties in odometery. State-of-the-art works in ball catching on mobile robots are accomplished on DLR's *Rollin' Junstin* robot [17], [18], [19], which senses the ball state with a pair of cameras mounted on the sides of head. It tracks two flying balls simultaneously with a multiple hypothesis approach. The UKF based estimator takes into account the sensor uncertainties rather than simply fitting a parabola curve. To deal with the kinematics uncertainties, such as elasticity of links and wheel slip, the robot computes its head pose based on inertial measurement unit (IMU) integration, instead of odometery. One interesting related work recently was done by Huang et al. that they implemented an RL controller to allow quadrupedal robots to hit balls as a football goalkeeper [20]. One major limitation is that ball tracking relies on an external camera instead of being onboard.

To summarize, this research presents three key challenges: (1) to holistically address noisy odometry, motion instability, and complex floating-base dynamics; (2) successfully catching a flying ball within a short period like one second requires the precise timing and accuracy of all sub-modules (simultaneous tracking and running-to-catch, and a close-loop catching strategy is a must for higher accuracy); and (3) using onboard sensors and computing present additional challenges, requiring the perception, localization, and control to function solely on a floating base robot itself.

B. Contribution

This study aims to achieve ball-catch tasks for a quadrupedal robot using only onboard sensors during dynamic walking. Legged robots, unlike wheeled counterparts, face numerous hard constraints and uncertainties, such as the constant discrete switch of foot-ground contact, foot slippage, and swing-stance landing impact. To successfully catch a flying ball during walking, it requires not only the prediction of the trajectory of a flying ball, but also a reliable state estimation to coordinate with dynamic gait generation as an integrated framework.

The contributions of this paper are as follows:

1) The design of an onboard vision system that embeds a stereo camera to track a custom-built infrared (IR) ball robustly and precisely, which allows the focused development of the state estimation.
2) A novel state estimation strategy that simultaneously estimates the positions of the ball and robot by fusing sensor information, which improves tracking and localization accuracy and resolves the challenge of noisy odometry during legged locomotion.
3) A control pipeline, comprising vision detection and dynamic locomotion control, is developed for quadrupedal robots with onboard sensors only to successfully perform dynamic ball-catching. To the best of our knowledge, this is the first such pipeline reported in the literature.

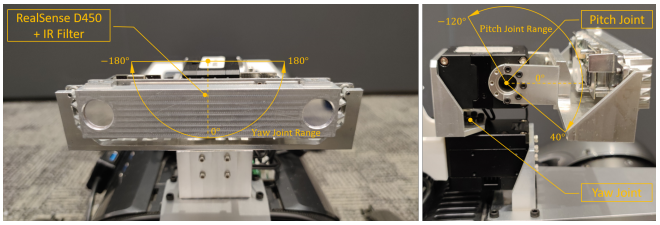


Fig. 2: The modified RealSense D450 camera with a cover and two IR filters mounted on the top of a 2-DoF head.

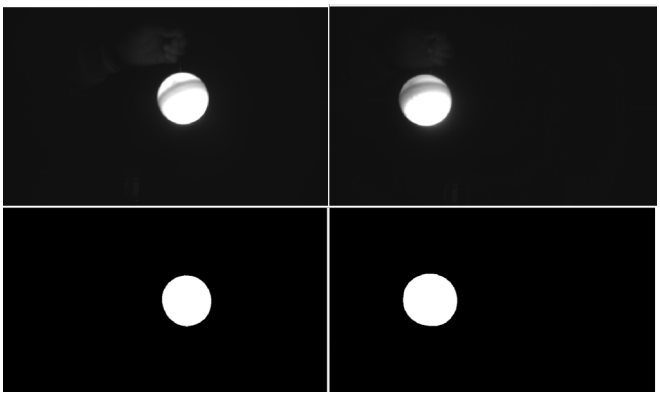


Fig. 3: The first row are the raw images from left and right cameras with IR bandpass filters and the second row are the corresponding binarized images. The ball was hanging in the air during day time but still well segmented.

This paper is organized as follows. Section II introduces the robot platform and the design of the vision system and IR ball. Section III elaborates on each module in the control pipeline. Section IV presents the experiments on the real robot and the data analysis. We conclude this study and suggest an intriguing direction of using event cameras for future work in the last section.

II. Hardware & System

Before introducing the control method, we first give a broad view about the robot and the modified vision system we used to complete this ball-catch task.

A. Robot Platform

The robot plaform we used to evaluate our algorithm is a quadrupedal robot named *CyberDog*, which is the first commercial legged robot product developed by Xiaomi Robotics Lab. There are totally 12 DoFs excluding the two of camera head, 3 for each leg, actuated by powerful quasi-direct drive motors. The onboard perception system contains various sensors, including touch sensor, ultra-sonic sensor, GPS module, an ultra-wide-angle fisheye lens, and Intel's RealSense D450 camera for depth-sensing. These components enable the robot to sense and communicate with surrounding environment. There are also one IMU module at the centre of body and encodes within each joint to determine the current configuration of robot. One external x86 computer (Intel CPU I7-8850H, 2.6Ghz to 4.3Ghz) is mounted on the back to replace its own Nvidia Xavier perception board for more computation power which guarantees better control stability

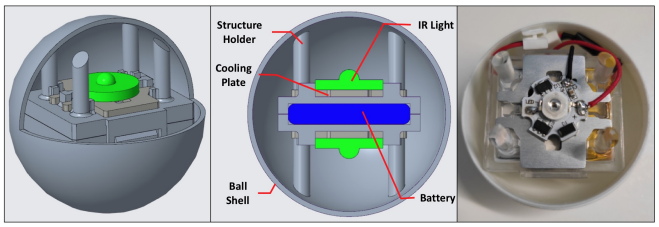


Fig. 4: IR light emission ball comprised of five parts: ball shell, structure holder, IR light, cooling plate, and battery. Left: isometric drawing. Middle: section view. Right: interior of the real ball.

and high communication bandwidth for data visualization and logging. A 3D printed plastic box was designed to hold the x86 PC which also contains a Lithium battery at the bottom to power up the PC.

B. Vision System

To extend the field of view (FoV), we moved the RealSense D450 camera from the front of *CyberDog* to the top of a 2-DoF head as shown in Fig. 1. The two DoF, pitch and yaw joints, are actuated by Dynamixel motors (XM430-W210-T) separately. The mechanical range of the yaw joint is $-180° \sim 180°$ while the pitch joint ranges from $-120°$ to $40°$. In addition, IR bandpass filters are added on the two infrared cameras of RealSense D450 so that the ball can be efficiently segmented out from surrounding environments via binarization as shown in Fig3. This dramatically improved the tracking robustness and accuracy.

C. Ball Design

We designed a ball with active IR light emission so that it can be easily captured by the aforementioned perception system. The design details are given in Fig. 4. The ball shell in the outer layer is separated into two pieces and screwed together when in use. It protects the whole structure from damage of impact and distributes the IR light evenly which is important for the binocular cameras to get the accurate ball centre. Similarly, the structure holder is also divided into two symmetric plastic parts and connected by snaps. In its core lies the lithium battery which powers up two IR lights (wavelength 850nm) on the upper and lower sides.

III. Methodology

In this section, we elaborate on the four main modules to achieve the ball-catch motions, namely the ball tracking module of stereo camera, simultaneous state estimation of both the ball and robot, ballistic trajectory prediction and locomotion control of quadrupedal robots. The whole control framework and the relationship among these modules are depicted in Fig. 5.

A. Ball Tracking

The ball tracking module is based on the binocular vision positioning technology. In order to obtain accurate image information, we performed stereo rectification and undistortion on the images according to the cameras' intrinsic and extrinsic parameters via ETH's kalibr tool [21]. With the help of extra IR bandpass filters, it is straightforward to segment out the target ball through the binarization technique of image processing as shown in Fig. 3. Thereafter, the coordinates of ball centres in the image can be obtained by calculating the centroid of the ball area. Considering possible image noise, the connected domain with largest area is chosen as the target ball area. Given the coordinates of ball centre in the left and right images, the 3-dimensional(3D) position of the ball in the camera frame [1] can be calculated via the principle of binocular parallax:

$$p^x = \frac{\left(u^l - c^x\right) p^z}{f^x}; p^y = \frac{\left(v^l - c^y\right) p^z}{f^y}; p^z = \frac{f \cdot b}{u^l - u^r}, \quad (1)$$

where p^x, p^y, p^z represent the ball position in the left image coordinate, which is right-handed with the positive y-axis pointing down, x-axis pointing right and z-axis pointing away from the camera. f is the focal length of the camera, b (also called the baseline) represents the distance between the optical centres of the two cameras, c^x, c^y, f^x, f^y are camera intrinsic parameters, and (u^l, v^l) are individually the horizontal and vertical pixel coordinates of the ball centre projected on the left camera while (u^r, v^r) for the right camera.

According to the specifications of RealSense D450, the depth accuracy is 2% at 4 meters range which is mainly decided by the physical limitation of camera resolution and the length of baseline. We got similar measurement accuracy with the proposed method when the target ball was held statically at certain distances. However, the detection of our approach is much more stable when the ball is flying in the air.

The FoV of D450's infrared cameras (Horizontal $87°$, Vertical $58°$) is so limited that the flying ball can easily get out of view and lose tracking. To address this issue, we mounted the camera on top of a 2-DoF head as shown in Fig. 2 so that the camera can move and keep staring at the target. A simple PD controller is implemented to achieve the camera view locking:

$$\begin{aligned} e_k^u &= \frac{u_k^l + u_k^r}{2} - u^c, e_k^v = \frac{v_k^l + v_k^r}{2} - v^c \\ q_k^{\text{yaw}} &= q_{k-1}^{\text{yaw}} + k_p \cdot e_k^u + k_d \cdot (\frac{e_k^u - e_{k-1}^u}{dt}) \\ q_k^{\text{pitch}} &= q_{k-1}^{\text{pitch}} + k_p \cdot e_k^v + k_d \cdot (\frac{e_k^v - e_{k-1}^v}{dt}) \end{aligned} \quad (2)$$

where q_k^{yaw} and q_k^{pitch} are the position commands individually sent to the motors of the head yaw and pitch joints, the subscript k and $k-1$ indicate the current and last time steps, u^c and v^c are the pixel coordinates of image horizontal and vertical centre, and k_p, k_d are the proportional and derivative coefficients which can vary between different joints. The very first joint commands q_0^{yaw} and q_0^{pitch} are given by reading the current position of motors. It should be noted that the actual position commands also consider the joint mechanical limits.

[1]Here we select the left camera coordinate as the camera frame.

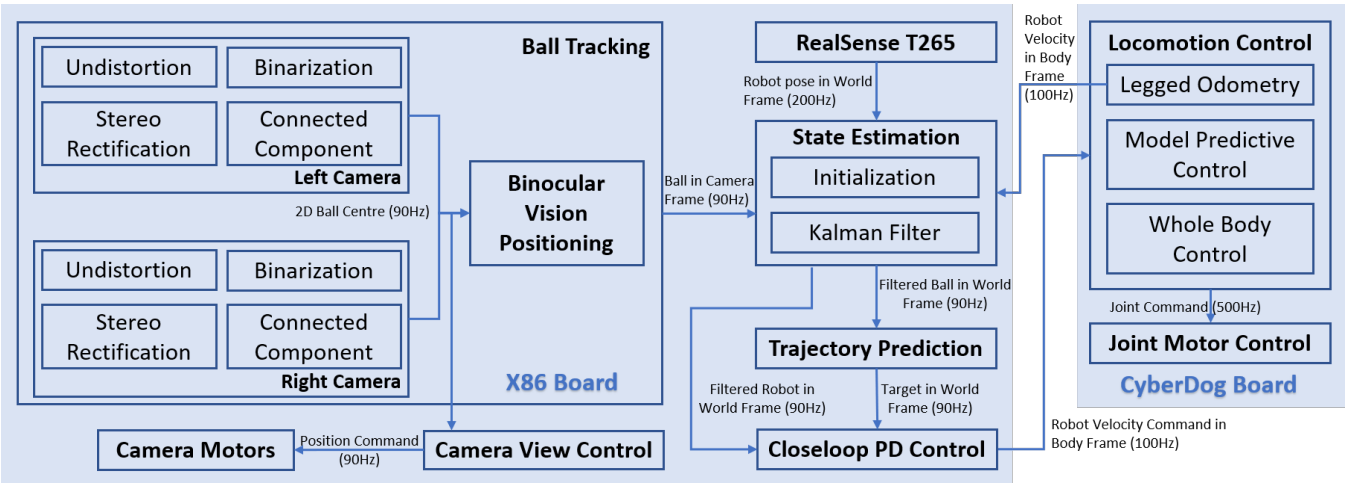


Fig. 5: Integrated control pipeline consists of four core parts: ball tracking, state estimation, trajectory prediction, and locomotion control. The ball tracking module estimates the 3D position of the target ball for the state estimation and trajectory prediction, and also produces the 2D ball centre in the images for the camera view control. The predicted goal position and estimated robot pose are the input to a closed-loop velocity feedback control to generate coordinated locomotion to chase the ball.

B. State Estimation

The ball tracking module acquires the ball position with respect to the camera. To catch the ball, we need to predict its trajectory in the future which requires the ball trajectory to be presented in an inertial coordinate, i.e., a world frame. By reading the motor status of the 2-DoF camera head, we can get the transformation from the camera frame to the robot frame. One missing link here is the robot pose with respect to the world. Unfortunately, CyberDog's legged odometry is not reliable enough, especially during fast locomotion with high step frequency. Compared with wheeled counterparts, this is a common pain point for legged robots due to frequent contact switches, foot slipping, and big landing impact. This makes the ball-catch task even more challenging in our case.

To alleviate this issue, we mounted a RealSense T265 module on the back for robot localization and implemented a Kalman filter taking both the ball position and robot position as the state for estimation. The idea is to correct the robot localization with the aid of a vision system and the known ballistic model of ball throwing trajectory. The corresponding state-space model can be written as:

$$\begin{bmatrix} \mathbf{X}^b \\ \dot{\mathbf{X}}^b \\ \mathbf{X}^r \end{bmatrix}_{k+1} = \begin{bmatrix} \mathbf{I} & \mathbf{T} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{X}^b \\ \dot{\mathbf{X}}^b \\ \mathbf{X}^r \end{bmatrix}_k + \begin{bmatrix} \hat{\mathbf{T}} & \mathbf{0} \\ \mathbf{T} & \mathbf{0} \\ \mathbf{0} & \mathbf{T} \end{bmatrix} \begin{bmatrix} \mathbf{G} \\ \mathbf{V}^r \end{bmatrix}_k \tag{3}$$

$$\begin{bmatrix} \mathbf{Z}^b \\ \mathbf{Z}^r \end{bmatrix}_{k+1} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{X}^b \\ \dot{\mathbf{X}}^b \\ \mathbf{X}^r \end{bmatrix}_{k+1} \tag{4}$$

where:

$$\begin{aligned} \mathbf{T} = dt \cdot \mathbf{I}, \quad \hat{\mathbf{T}} = \frac{1}{2} dt^2 \cdot \mathbf{I} \\ \mathbf{G} = \begin{bmatrix} 0 & 0 & -g \end{bmatrix}^T \end{aligned} \tag{5}$$

Eq. (3), (4) is the discretized state-space model we looked into, and the states $\mathbf{X}^b$, $\dot{\mathbf{X}}^b$, $\mathbf{X}^r$ are individually the ball position, ball velocity and robot position in the assumed world frame. Subscript k and $k+1$ indicates the time step. The input $\mathbf{G}$ is the gravity acceleration vector while $\mathbf{V}^r$ is the robot velocity in the world frame which can be calculated as: $\mathbf{V}^r = {}^w\mathbf{R}_r \mathbf{V}_r^r$. ${}^w\mathbf{R}_r$ is the rotation matrix from robot to world frame and $\mathbf{V}_r^r$ is the robot velocity represented in the robot frame which is given by the legged odometry. Similarly, the observation $\mathbf{Z}^b$ is the ball position relative to the robot represented in the world frame instead of camera frame which is done by transforming the measurement of ball tracking module with the head motor readings and ${}^w\mathbf{R}_r$. On the other hand, $\mathbf{Z}^r$ is the robot position in the world frame given by the RealSense T265. All the aforementioned variables are 3D vectors indicating the x, y, z directions. In the following of this paper, x is the forward direction our robot heads to, y represents the lateral direction and z the vertical direction. In Eq. (5), dt is the time interval between the k and $k+1$ time step, g is the scalar gravity acceleration, and $\mathbf{I}$ is an identity matrix with 3×3 dimension.

Following the canonical paradigm of Kalman fitler, the iterative steps of state estimation can be divided into two steps, *predict* and *correct*. To accelerate the estimation process, we provided a dedicate initialization strategy for the Kalman filter which is critical for fast motions like ball-catch. More details are given below.

1) Start: At the beginning of the ball-catch task, to initialize the Kalman filter, the robot stands in place and gazes at the ball first to have a steady view. After collecting certain frames of binocular camera images, the ball velocity is estimated according to the equations below:

$$\begin{bmatrix} \dot{x}_k \\ \dot{y}_k \\ \dot{z}_k \end{bmatrix} = (\mathbf{X}_s^b + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \frac{1}{2} g \mathbf{T}_s \odot \mathbf{T}_s \end{bmatrix}) \mathbf{T}_s^\dagger + \begin{bmatrix} 0 \\ 0 \\ -g(t_k - t_0) \end{bmatrix} \tag{6}$$

where:

$$\begin{aligned} \mathbf{T}_s &= \begin{bmatrix} t_1 - t_0 & t_2 - t_0 & \cdots & t_k - t_0 \end{bmatrix} \\ \mathbf{X}_s^b &= \begin{bmatrix} x_1 - x_0 & x_2 - x_0 & \cdots & x_k - x_0 \\ y_1 - y_0 & y_2 - y_0 & \cdots & y_k - y_0 \\ z_1 - z_0 & z_2 - z_0 & \cdots & z_k - z_0 \end{bmatrix} \end{aligned} \tag{7}$$

$[\dot{x}_k \ \dot{y}_k \ \dot{z}_k]^T$ is the latest estimated ball velocity along all three directions. The subscript $0, 1, \cdots, k$ indicates the window

size of $k+1$ ball positions we collected for the initialization. 0 stands for the first data while k is the latest. $t_0, t_1, \cdots, t_k$ records the timestamp of each ball position. The operators $\dagger$ and $\odot$ are individually the pseudo-inverse and the element-wise multiplication.

Given the latest ball position $\mathbf{X}_k^b$, the computed velocity $\dot{\mathbf{X}}_k^b$, and the robot state from RealSense T265, we can set up the initial state of Kalman filter accordingly.

2) Predict: According to Eq. (3), we can predict the next state ahead based on the current state and input:

$$\mathbf{X}_{k+1}^- = \mathbf{A}\mathbf{X}_k + \mathbf{B}\mathbf{u}_k + w_k \tag{8}$$

Eq. (8) is a simplified notation of Eq. (3) for the following elaboration where $\mathbf{X}_k = [\mathbf{X}^b \ \dot{\mathbf{X}}^b \ \mathbf{X}^r]_k^T$, $\mathbf{u}_k = [\mathbf{G} \ \mathbf{V}^r]_k^T$ and $\mathbf{A}$, $\mathbf{B}$ are the corresponding state and input matrices. The predicted next state $\mathbf{X}_{k+1}^-$ noted with the super minus is known as *a priori* state estimate at step $k+1$ given knowledge of the process prior to step $k+1$ [22]. w_k is a random variable representing the process noise which is assumed to be independent, white and with normal probability distribution $p(w) \sim N(0, \mathbf{Q})$. Hence, provided the process noise covariance $\mathbf{Q}$, the *a priori* estimate error covariance can be projected:

$$\mathbf{P}_{k+1}^- = \mathbf{A}\mathbf{P}_k\mathbf{A}^T + \mathbf{Q} \tag{9}$$

3) Correct: We assume there also exists an Gaussian noise v_{k+1} among the observation equation (4):

$$\mathbf{Z}_{k+1} = \mathbf{H}\mathbf{X}_{k+1} + v_{k+1} \tag{10}$$

where $\mathbf{Z}_{k+1} = [\mathbf{Z}^b \ \mathbf{Z}^r]_{k+1}^T$ and $\mathbf{H}$ is the corresponding observation matrix. v_{k+1} is also represented with a normal distribution with the measurement noise covariance $\mathbf{R}$: $p(v) \sim N(0, \mathbf{R})$.

Thereafter, we can compute the Kalman gain $\mathbf{K}_{k+1}$ based on measurement update and prediction in Eq. (9):

$$\mathbf{K}_{k+1} = \frac{\mathbf{P}_{k+1}^-\mathbf{H}^T}{\mathbf{H}\mathbf{P}_{k+1}^-\mathbf{H}^T + \mathbf{R}} \tag{11}$$

And the corrected *a posteriori* estimate state and error covariance are:

$$\begin{aligned} \mathbf{X}_{k+1} &= \mathbf{X}_{k+1}^- + \mathbf{K}_{k+1}(\mathbf{Z}_{k+1} - \mathbf{H}\mathbf{X}_{k+1}^-) \\ \mathbf{P}_{k+1} &= (\mathbf{I} - \mathbf{K}_{k+1}\mathbf{H})\mathbf{P}_{k+1}^- \end{aligned} \tag{12}$$

C. Trajectory Prediction

In the ball-catch task, the robot needs to move in advance before the ball reaching. This requires the prediction of goal position according to the existing ball trajectory. Considering the weight and size of the ball we used, the influence of aerodynamics is negligible when it flies in the air. Therefore, ballistic model is taken for the prediction:

$$\begin{aligned} z(t) &= z_k + \dot{z}_k t - \frac{1}{2}gt^2 = h \\ t &= \frac{\dot{z}_k + \sqrt{\dot{z}_k^2 + 2g(z_k - h)}}{g} \end{aligned} \tag{13}$$

where z_k and $\dot{z}_k$ are the current ball position and velocity along the vertical direction, and h is the desired height which should be set as the same height of the basket opening for ball-catch. The computed t gives the remaining time for the ball reaching to the goal. Because the velocities along x and y directions are constant in the ballistic model, we can easily get the goal position on the horizontal plane as:

$$\begin{aligned} x^{\text{goal}} &= x_k + \dot{x}_k t \\ y^{\text{goal}} &= y_k + \dot{y}_k t \end{aligned} \tag{14}$$

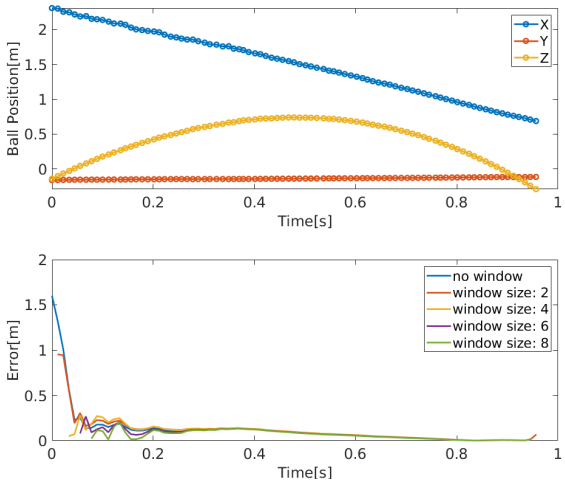


Fig. 6: Experiment result of static testing. The upper plot is the ball tracking position along x, y, z directions over time, and the lower plot is the prediction error over time with different initialization window sizes of Kalman filter.

D. Locomotion Control

To make sure the robot arrives at the target place accurately, we implemented a close-loop PD controller to determine the desired locomotion velocities:

$$\begin{aligned} \mathbf{V}_w^{\text{cmd}} &= \mathbf{k}_p(\mathbf{X}_w^{\text{goal}} - (\mathbf{X}_w^r + {}^w\mathbf{R}_r\mathbf{L})) + \mathbf{k}_d{}^w\mathbf{R}_r\dot{\mathbf{X}}_r^r, \\ \mathbf{V}_r^{\text{cmd}} &= {}^r\mathbf{R}_w\mathbf{V}_w^{\text{cmd}}, \end{aligned} \tag{15}$$

where $\mathbf{V}_w^{\text{cmd}}$ and $\mathbf{V}_r^{\text{cmd}}$ are individually the robot velocity commands in the world and robot frames. They are 2-dimensional(2D) vectors standing for the velocities along x and y directions. Accordingly, $\mathbf{X}_w^{\text{goal}} = [x^{\text{goal}} \ y^{\text{goal}}]^T$ is the 2D goal position and $\mathbf{X}_w^r$ is the current position of robot in the world frame. L is the relative position between basket centre and robot centre in the robot frame. $\dot{\mathbf{X}}_r^r$ is the current robot velocity in the robot frame measured by the legged odometry. ${}^w\mathbf{R}_r$ and ${}^r\mathbf{R}_w$ are 2×2 orientation matrices of yaw rotation between robot and world. $\mathbf{k}_p$, $\mathbf{k}_d$ are the proportional and derivative coefficients. The final velocity commands are clamped by certain limits to maintain a stable and safe motion. In addition, the angular velocity of yaw motion is set to zero and the desired body height is constant.

The locomotion control algorithm we adopted is similar to the work of MIT Biomimetics Lab [23]. Given desired locomotion velocities, model predictive control(MPC) plans the body trajectory and reference contact force while whole body controller is in charge of tracking planned trajectories of MPC and generates joint commands accordingly. The joint commands including desired position, velocity and torque are served by joint motor controllers running at 20kHz.

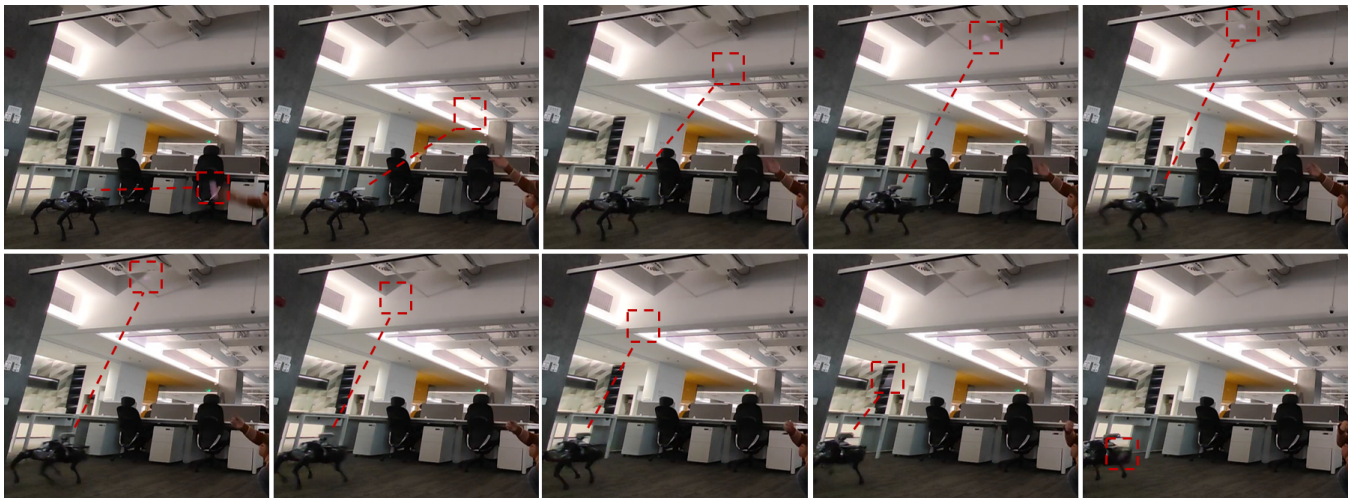


Fig. 7: Example of CyberDog's actual catch. The snapshots show the process from throwing the ball to the robot running and catching the ball. The time interval between two neighboring snapshots is 0.13s. The upper left corner is the initial state, and the lower right corner is the final state. The ball is highlighted with a red dashed box and the dash line indicates the direction of camera view.

IV. Experiment

Several experiments were conducted on our robot *CyberDog* to evaluate the performance of the proposed approach. First, we carried out static testing to verify the accuracy and robustness of the detection and prediction module without involving the noise of robot localization under movement. Then, actual ball-catch experiments were performed to show the validity of the proposed method. The success rate of the ball-catch task is more than 80% when the ball falls into the reachable area (2x1.5m square). Fig. 7 shows a typical example of a successful catch.

A. Static Testing

In this experiment, the robot lied on a table without moving, and the ball was thrown to the front side of the robot. As shown in the upper plot of Fig. 6, the ball tracking trajectories are quite smooth except that the depth(x) direction was a bit shaky when the ball was more than 2 meters away from the camera. This is acceptable considering the physical limitation of the camera as mentioned in Section III-A.

The lower plot of Fig. 6 presented the prediction error over time when using different initialization window sizes for the Kalman filter in Eq. (6). The prediction error was calculated based on 3D Euclidean distance by comparing the prediction result with the last detected ball position at the end of the trajectory, which was taken as the ground truth because it was closer to the camera with less measurement error. Meanwhile, the prediction height h in Eq. (13) was also set the same as the last ball position. The result indicated that we can get a better prediction earlier if choosing a proper initialization window size.

B. Dynamic Catch

As shown in Fig. 7, the ball was thrown around 2 meters away from the robot and the whole flying trajectory lasted for roughly 1.2 seconds in this experiment. During the period, the robot stayed in place first to get a steady view of the ball and then ran to its right behind to catch the ball (diameter 6cm) with a basket (diameter 14cm) mounted in the front.

Fig. 8 presents the recorded 3D tracking position of the ball and the robot. The red dot stands for the measured ball position which simply added the camera tracking result, motor readings of 2-DoF camera head, and RealSense T265 output together. Obviously, there are some jumpy and unrealistic points in the middle. On the contrary, the ball position marked as a blue dot is much more stable and smoother after going through the state estimator. The predicted goal is labeled as a green dot which correctly fell into the basket opening. This is consistent with the snapshots in Fig. 7.

The tracking data shown in Fig. 9 gives more details about this experiment. Subplot 9a shows the ball tracking results along all directions. The blue cycle line is the directly measured position while the red dot line stands for the one after Kalman filter. The window size k in Eq. (6) is set to 10 for the initialization. The trajectories fit quite well to the ballistic model, i.e., the movements are linear along horizontal x and y directions and parabola along the vertical z direction.

Subplot 9b presents the prediction errors along x, y directions which are calculated by comparing the last prediction with others over time. The last prediction is taken as the ground truth because the time it happened was closest to the real catch whose result was verified by the experiment. The z direction is constant and set according to the height of the basket opening, e.g., 0.386m in this case. The prediction error changed over time and converged to a reasonable value around 0.3s after the ball was thrown. The spike of x direction around 0.2s is because the robot started to move at that time and added more noise to the robot localization. Compared with the x direction, the smaller prediction error of the y direction is due to the small lateral velocity of the flying ball.

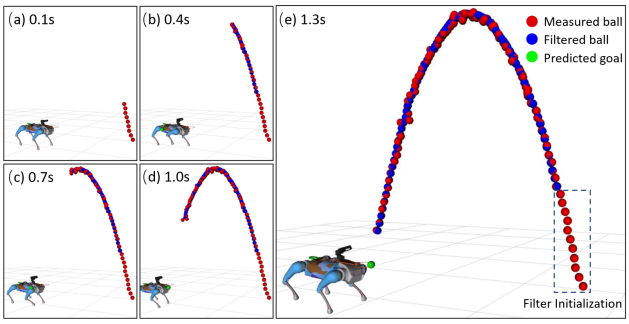


Fig. 8: Visualization of the ball-catch experiment via Rviz2. The time interval between two neighboring subplots is 0.3s.

V. Conclusion

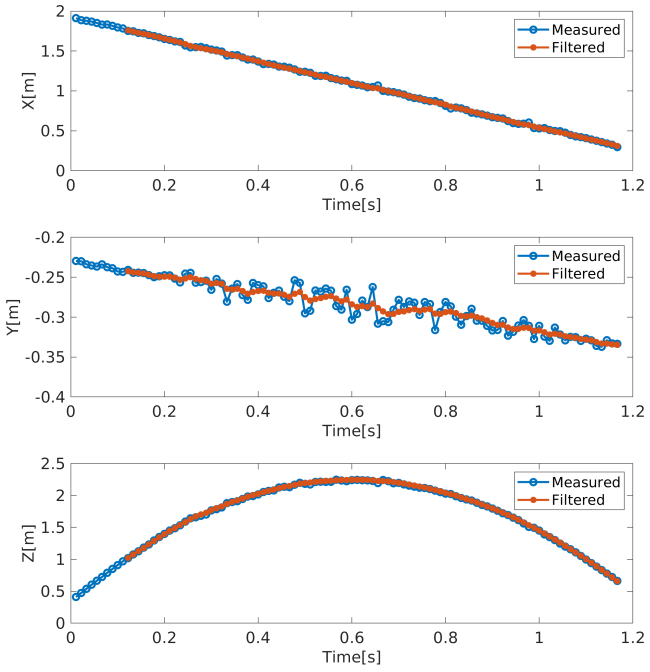
This paper proposed a control pipeline from stereo vision to legged locomotion which allows quadrupedal robots to run to catch flying balls. First, to achieve stable, accurate and high-frame-rate tracking onboard, we designed an infrared ball and located the flying ball based on binocular vision positioning with onboard cameras by using IR filters. The stereo camera on top of a 2-DoF head enlarges the camera view. Thereafter, to resolve the significant noise of legged odometry, a RealSense T265 on the robot's back was used for implementing a Kalman filter to estimate both the ball and robot position simultaneously, by fusing the vision positioning, camera motor readings, T265 localization, and legged odometry altogether. In addition, we provided an initialization strategy for the Kalman filter in case acceleration is required for earlier prediction in dynamic tasks. The proposed approach was validated well in static testing and demonstrated successful dynamic catch experiments on the real *CyberDog* robot.

One limitation is that to ensure the sole use of onboard sensing and computing for the state estimation, we designed an active IR emitting ball to bypass the processing speed limitation stemming from the current vision solutions. Therefore, to overcome the underlying issue in the vision system, a completely new hardware solution is worth investigating. For example, using an event camera to detect and extract the outline of fast-moving objects at hundreds of Hz – matching the frequency of servo control loops.

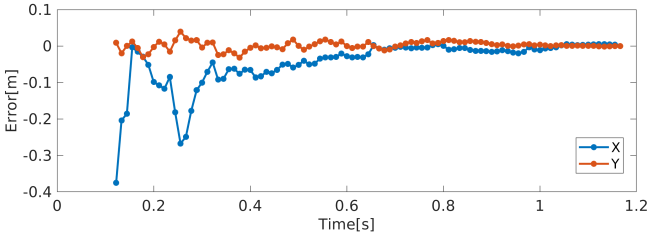
For this, we believe that future studies can build on or reuse our proposed control framework and implement event cameras for flying object detection and tracking, thus significantly augmenting the capability for real applications. Though given the limited resources and obligations within a research project, we were not able to pursue this direction. Nonetheless, it presents an intriguing direction for the research community to explore further.

Acknowledgment

This work is supported by Xiaomi Robotics Lab. Special thanks to mechatronic engineers Yichu Yang, Peng Yu and Jian Xiao for their help with the system design.



(a) Ball position tracking along x, y, z directions over time



(b) Prediction error along x, y directions over time

Fig. 9: Ball position tracking and prediction errors during the ball-catch experiment.

References

[1] R. Mori, K. Hashimoto, and F. Miyazaki, "Tracking and catching of 3d flying target based on gag strategy," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 5. IEEE, 2004, pp. 5189–5194.

[2] K. Deguchi, H. Sakurai, and S. Ushida, "A goal oriented just-in-time visual servoing for ball catching robot arm," in *2008 IEEE/RSJ International conference on intelligent Robots and Systems*. IEEE, 2008, pp. 3034–3039.

[3] G.-R. Park, K. Kim, C. Kim, M.-H. Jeong, B.-J. You, and S. Ra, "Human-like catching motion of humanoid using evolutionary algorithm (ea)-based imitation learning," in *RO-MAN 2009-The 18th IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 2009, pp. 809–815.

[4] V. Lippiello, F. Ruggiero, and B. Siciliano, "3d monocular robotic ball catching," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1615–1625, 2013.

[5] S. Kim, A. Shukla, and A. Billard, "Catching objects in flight," *IEEE Transactions on Robotics*, vol. 30, no. 5, pp. 1049–1065, 2014.

[6] K. M. Lynch and M. T. Mason, "Dynamic nonprehensile manipulation: Controllability, planning, and experiments," *The International Journal of Robotics Research*, vol. 18, no. 1, pp. 64–92, 1999.

[7] W. Hong and J.-J. E. Slotine, "Experiments in hand-eye coordination using active vision," in *Experimental Robotics IV*. Springer, 1997, pp. 130–139.

[8] U. Frese, B. Bauml, S. Haidacher, G. Schreiber, I. Schäfer, M. Hahnle, and G. Hirzinger, "Off-the-shelf vision for a robotic ball catcher," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the*

the Next Millennium (Cat. No. 01CH37180), vol. 3. IEEE, 2001, pp. 1623–1629.

[9] J. Kober, K. Muelling, and J. Peters, "Learning throwing and catching skills," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5167–5168.

[10] J. Kober, M. Glisson, and M. Mistry, "Playing catch and juggling with a humanoid robot," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE, 2012, pp. 875–881.

[11] M. Müller, S. Lupashin, and R. D'Andrea, "Quadrocopter ball juggling," in *2011 IEEE/RSJ international conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 5113–5120.

[12] P. Bouffard, A. Aswani, and C. Tomlin, "Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 279–284.

[13] R. Silva, F. S. Melo, and M. Veloso, "Towards table tennis with a quadrotor autonomous learning robot and onboard vision," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 649–655.

[14] K. Su and S. Shen, "Catching a flying ball with a vision-based quadrotor," in *International Symposium on Experimental Robotics*. Springer, 2016, pp. 550–562.

[15] S.-T. Kao, Y. Wang, and M.-T. Ho, "Ball catching with omni-directional wheeled mobile robot and active stereo vision," in *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*. IEEE, 2017, pp. 1073–1080.

[16] S.-T. Kao and M.-T. Ho, "Ball-catching system using image processing and an omni-directional wheeled mobile robot," *Sensors*, vol. 21, no. 9, p. 3208, 2021.

[17] B. Bäuml, O. Birbach, T. Wimböck, U. Frese, A. Dietrich, and G. Hirzinger, "Catching flying balls with a mobile humanoid: System overview and design considerations," in *2011 11th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2011, pp. 513–520.

[18] O. Birbach, U. Frese, and B. Bäuml, "Realtime perception for catching a flying ball with a mobile humanoid," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 5955–5962.

[19] B. Bäuml, F. Schmidt, T. Wimböck, O. Birbach, A. Dietrich, M. Fuchs, W. Friedl, U. Frese, C. Borst, M. Grebenstein *et al.*, "Catching flying balls and preparing coffee: Humanoid rollin'justin performs dynamic and sensitive tasks," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3443–3444.

[20] X. Huang, Z. Li, Y. Xiang, Y. Ni, Y. Chi, Y. Li, L. Yang, X. B. Peng, and K. Sreenath, "Creating a dynamic quadrupedal robotic goalkeeper with reinforcement learning," Oct. 2022.

[21] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1280–1286.

[22] G. Welch, G. Bishop *et al.*, "An introduction to the kalman filter," 1995.

[23] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *arXiv preprint arXiv:1909.06586*, 2019.