PetBot: Your Commanded Companion

1st SHIUAN-JEN YANG
Dept. Electrical Engineering, National Taiwan University
Taipei, Taiwan (R.O.C.)
Email: r12921013@ntu.edu.tw

2nd FENG-EN YEH
Dept. Mechanical Engineering, National Taiwan University
Taipei, Taiwan (R.O.C.)
Email: b08502091@ntu.edu.tw

3rd YU-HSIN CHEN
Dept. Electrical Engineering, National Taiwan University
Taipei, Taiwan (R.O.C.)
Email: r12921009@ntu.edu.tw

4th YU-HAN TSAI
Dept. Mechanical Engineering, National Taiwan University
Taipei, Taiwan (R.O.C.)
Email: d12522006@ntu.edu.tw

5th FENG-YI CHEN
Dept. Electrical Engineering, National Taiwan University
Taipei, Taiwan (R.O.C.)
Email: r12921010@ntu.edu.tw

***Abstract*—This study attempts to address the growing societal challenges associated with loneliness, especially prevalent among young individuals leading childless or unmarried lifestyles, and the elderly facing dementia. Leveraging the Pioneer 3-DX robot, equipped with RealSense D435i Camera, we introduce PetBot—a robotic companion designed to mitigate loneliness. The study combines gesture recognition and vision-based distance control, enhancing PetBot's responsiveness and interaction with users. The methodology involves utilizing pre-trained toolkits, OpenCV, mediaPipe, and cvzone, for object detection, tracking, and gesture recognition. PetBot's functionalities include tracking, stopping, happy reactions, spinning, and playing tennis ball. Distance estimation integrates MediaPipe, OpenCV, and Pyrealsense, ensuring accurate and stable tracking. Gesture recognition employs custom gestures, minimizing noise impact through consecutive frame analysis. Tennis ball localization uses HSV color space for robust detection. The study achieves real-time communication between the vision and robot programs, showcasing PetBot's integration and feedback control. Experimental results demonstrate PetBot's ability to track, react, and play, emphasizing the potential impact on loneliness alleviation and cognitive health.**

***Index Terms*—companion, Pioneer, gesture recognition, Open CV**

I. Introduction

In modern society, the proportion of childless or no marriage lifestyle among young people has largely increased. Due to the busy work lifestyle, they could probably not have time to keep and take care of pets. This may cause them to feel depressed and lonely and bring negative influence to their mental health.

As the extension of human lifespan, the portion of ageing population is increasing. Among elderly, dementia has become a critical issue. As the Alzheimer's Disease (AD) is the leading cause of dementia. Isolation and loneliness have been identified among the major risk factors for AD. The increasing prevalence of both loneliness and AD emphasizes the urgent need to understand this association to inform treatment. According to medical research, language or gesture communication can activate brain, which can reduce the speed of becoming dementia.

Pioneer 3-DX is a small lightweight two-wheel two-motor differential drive robot ideal for indoor laboratory or classroom use. Due to versatility, reliability and durability have made them the preferred platform for advanced intelligent robotics. It's one of the world's most popular intelligent mobile robots for education and research. Pioneer 3-DX are pre-assembled, customizable, upgradeable, and rugged enough to last through years of laboratory and classroom use.

The main contributions of this article are as follows:

(1) Realize PetBot, which can behave like a pet to do the demanding task based on the instructions of human, by combining gesture recognition with Pioneer 3-DX mobile robot and RealSense D435i Camera.

(2) Train the gesture recognition vision detection model by ourselves.

Code is available at https://github.com/Florrie111/PetBot and the demo video is available at https://shorturl.at/dentI.

II. Related Work

In this section, we first reviews the computer vision technologies in II-A and II-B and briefly introduce the concept and applications. Then, we also discuss the relevant research conducted in the operation of Pioneer P3-DX in II-C.

A. Open Source Computer Vision

Open Source Computer Vision (OpenCV) is an expansive open-source library that includes hundreds of computer vision algorithms [1]. Developed with the aim of establishing a unified infrastructure for diverse computer vision applications, OpenCV plays a crucial role in fostering the integration and acceleration of machine perception within commercial products.

B. MediaPipe

MediaPipe is an open-source framework for building machine learning pipelines for processing time-series data, encompassing various modalities such as video, audio, and more [2]. Specifically crafted for computer vision pipelines, MediaPipe incorporates a spectrum of components integral to its functionality, comprising model inference, sophisticated media processing algorithms, and intricate data transformations. By seamlessly integrating these elements, MediaPipe facilitates the development of highly efficient and adaptable systems capable of extracting meaningful insights from dynamic, real-world data streams.

C. Pioneer Operation

The integration of the Pioneer P3-DX platform has been pivotal in advancing autonomous robotics research, as evidenced by notable works. In an exploration of hazard environments, an Autonomous Scouting Robot (ASR) operates dynamically in three phases: scene fetching, human discovery, and following firefighters for rescue, showcasing the platform's adaptability [3]. A motor velocity-based multi-objective genetic algorithm facilitates obstacle avoidance for the Two-Wheeled Pioneer P3-DX Robot, as demonstrated through successful simulation tests [4]. Behavior-based robotic algorithms on the Pioneer P3-DX in a maze exploration mission highlight its ability to autonomously trace targets, utilizing wall-following behavior and PID control for precise navigation [5].

III. Methodology

A. System Overview

The PetBot system integrates vision-based capabilities and robotic control to create an interactive and responsive robotic companion.

In III-B, we utilize some off-the-shelf pre-trained toolkits to extract the posture, gesture, and the other information from images which are captured by RealSense D435i. The distance estimation process combines MediaPipe, OpenCV, and Pyrealsense, using a self-designed method for accurate measurements. The flow chart of the proposed system is shown in Figure 1.

In III-C, the data transmission system ensures real-time communication between the image processing and control programs.

In III-D, we use RealSense D435i data to keep the owner centered and ensure PetBot maintains a set distance by P control.

B. Vision

To enhance the perceptual capabilities of PetBot and facilitate its interaction with the environment, we leverage off-the-shelf pre-trained toolkits such as OpenCV and mediaPipe for object detection and tracking.

The visual processing workflow is illustrated in Figure 1. Initially, RGB-D frames are captured using the RealSense D435i, respectively. Subsequently, image processing is employed for tasks including distance estimation, gesture recognition, and tennis ball localization. In the absence of detected individuals in the images, PetBot enters a searching mode, executing a spinning action while searching for its owner. Upon the appearance of a person, PetBot adjusts its behavior based on the recognized gestures made by the owner. Crucially, pertinent information such as the current operational mode, directional turning (left or right), and the horizontal and lateral distances from humans or objects is systematically recorded and stored in a text file for future reference.

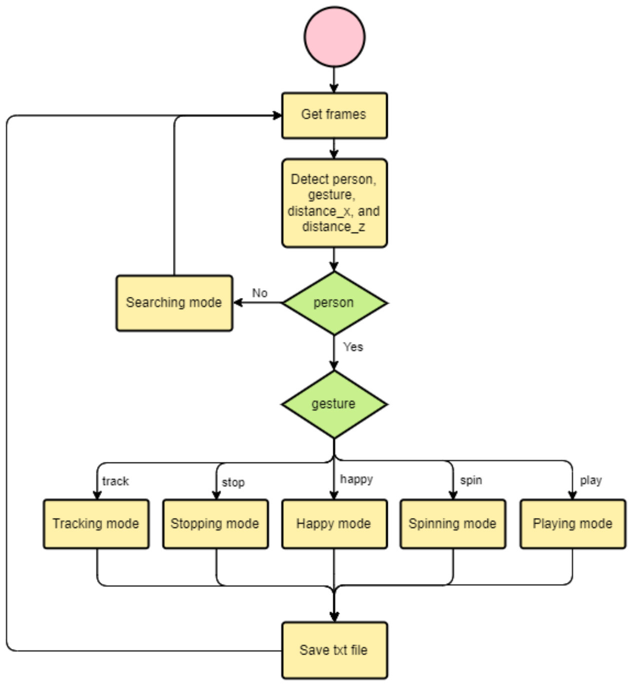

Fig. 1. The flow chart of vision section.

1) Distance estimation: The distance information is required in tracking mode. To estimate the distance between the Petbot and the owner, several tools and package are used. MediaPipe provides a Machine Learning solution for body pose recognition and capturing. Opencv is used in the image processing tasks. Pyrealsense is utilized with RealSense D435i camera, offering precise distance estimation. In order to avoid collision with the owner, horizontal distance should be calculated correctly. However, some challenges occurs because of the limited vision of the camera and the couple of vertical and horizontal distance. To address this problem, a self-designed method is presented. The implementation of the method is as followed: 1) The x-y position of body pose landmark such as nose, left shoulder, right shoulder, left hip, and right hip in the image can be derived by using Mediapipe. 2) Center of the body can be obtained by taking average of x, y coordinate from the five landmarks derived in last step. 3) Measure the real-world distance of the center of body landmark by PyrealSense. 4) Apply Trimmed mean method in Equation (1) by averaging data after removing the maximum and minimum data. where $\bar{x}$ is the average value; x is the sorted array which consists of samples x_i.; N is the number of samples. 10 samples are used. 5) Implement Exponential Moving Average (EMA) in Equation (2) to smooth the distance estimation.

$$trimmed\,Mean = \bar{x} = \frac{1}{N-2}\sum_{i=n+1}^{N-2} x_i \quad (1)$$

$$S_t = \alpha \times Y_t + (1-\alpha) \times S_{t-1} \quad (2)$$

where α is the soothing factor; S_{t-1} is the measurement from last state; Y_t is the measurement obtained currently. In the subsequent process, the distance estimation is updated in real-time and transmitted to the Petbot control program for further distance control tasks. This method prioritizes smoothness and consistency as the utmost considerations to ensure tracking stability.

2) Gesture Recognition: Gesture recognition plays a pivotal role in adjusting PetBot's actions based on the owner's instructions. Leveraging the cvzone packages from the Github repository [6], we employ the pre-trained HandDetector toolkit to detect and recognize gestures. This toolkit provides essential information, such as landmarks, centers, bounding boxes, and the number of fingers raised, for one or multiple hands. To establish custom gestures for distinct actions, we utilize the $fingersUp$ function within HandDetector, assessing the upward extension of each finger. Our self-defined gestures, illustrated in Figure 2, encompass six modes: spinning, tracking, happy, stopping, playing, and stopping playing. However, during PetBot's movement, relying on a single frame for mode determination may yield unstable and unreliable results. Rapid mode switches can occur due to unexpected noises. To mitigate this issue, we implement a counter to assess consecutive frames for consistent gestures. If the same gestures persist for three frames or more, we convert to another mode corresponding to the recognized gesture. This strategy minimizes the impact of noise and recognition errors, enhancing the system's robustness.

3) Tennis Ball Localization: In order to facilitate PetBot to chase the tennis ball during the playing mode, the implementation of tennis ball localization becomes crucial. In this part, we employ Opencv functions for image processing to detect the tennis ball. Leveraging the distinctive brightness of the tennis ball, we apply the HSV color range as a threshold, designating the bright yellow segment as white and the remainder as black. Subsequent erosion and dilation operations are performed to refine the boundary and remove the noises. Following this, contours of the white area, which is bright yellow area in the original image, are identified, and the center of the tennis ball is calculated by utilizing momentum. Consequently, we obtain the x and y coordinates in the image, enabling determination of whether PetBot should turn left or right to chase the tennis ball or follow the owner. Assuming the width and height of an image are denoted as W and H, with x and y representing the coordinates, a threshold value a is established to determine the turning direction.

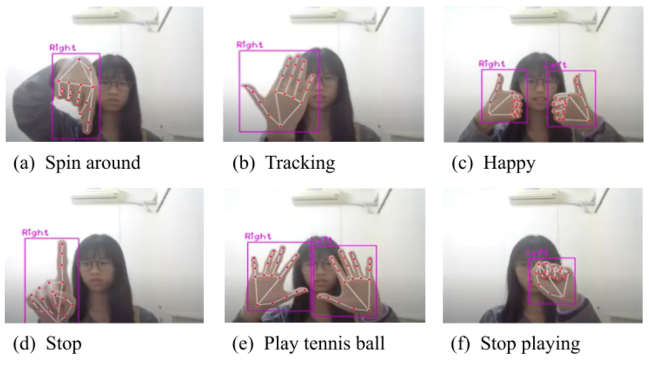

Fig. 2. The self-definite gestures. By recognizing these gestures, PetBot will enter (a) spinning mode and turn 360 degrees. (b) tracking mode and follow its owner. (c) happy mode and swing to the left, swing to the right and turn 360 degrees. (d) stopping mode and wait in the current place. (e) playing mode and start to chase the tennis ball. (f) stopping playing mode and go back to stopping mode.

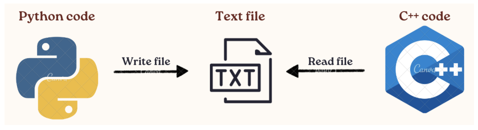

Fig. 3. The concept to fulfill data transmission.

$$turn = \begin{cases} 1, & \text{if } (x - \frac{W}{2} > a) \\ -1, & \text{if } (x - \frac{W}{2} < -a) \\ 0, & \text{else} \end{cases} \quad (3)$$

In Equation (3), a value of 1 for $turn$ implies a right turn, -1 denotes a left turn, and 0 signifies moving straight ahead. However, challenges arise when the tennis ball temporarily disappears from the frames, which is a common scenario during when playing ball with real dogs. For instance, when we throw the tennis ball far away, a real dog may still accurately track its trajectory. To address this issue, we retain the x and y coordinates from the last frame where the tennis ball was visible. This ensures that PetBot continues turning left or right when the ball vanishes from either side. This adaptive adjustment enhances PetBot's behavior, mimicking the responsiveness of a real dog during a game of chasing a ball with a human.

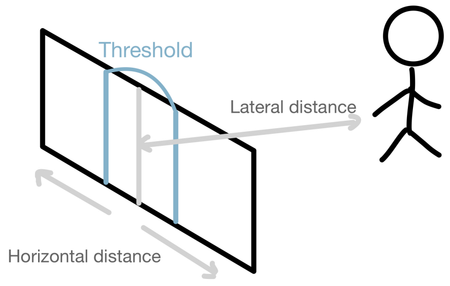

Fig. 4. Demonstration of distance identification

C. Data Transmission

In this work, the image processing is executed by Python on the host machine, while the control program runs on the virtual machine using C++. The data derived by the image processing such as owner's distance, ball distance needs to be transmitted from the host machine to virtual machine. Therefore, a common text file is generated as a platform enable publication from the image processing and subscription for the Petbot control program. The concept to fulfill data transmission is shown in Figure 3. The text file is located at the shared folder to enable access for image processing and the Petbot control program.

D. Robot

The tasks of PetBot include 5 different modes: Tracking Mode, Stopping Mode, Happy Mode, Spinning Mode and Playing Mode. According to different gestures from the owner, PetBot demonstrates different modes.

1) Feedback Control (Mode Tracking and Playing): PetBot identifies the depth distance and the horizontal location of the target ($owner$ or $ball$)with RealSense D435i camera. Then we apply feedback control on distance inputs to have PetBot maintain a fixed distance from the target and also keep the owner in the middle of camera sight. As for the feedback control, we implemented P control, and set the input as the distance from the desired location with the output as the rotational speed of the wheels. The output is calculated as lateral speed and pure rotational speed respectively, while the lateral speed ensures PetBot keeps a fixed distance from the target and pure rotational speed ensures the target is in the middle of the camera sight.

```
void PID(ArRobot *robot, double ownerLatPos,
            double ownerTurnPos, int turn)
{
  double _turnPos = turnPos;
  if(turn == 0) _turnPos = 0;

  double MaxSpeed;
  double MaxRotSpeed;
  double desiredLatPos;
  double desiredTurnPos = 0;

  /****Parameter****/
  static double _dPos_err, _dTurn_err;
  static double latSpeed, rotSpeed;

  /****Lateral****/
  _dPos_err = latPos - desiredLatPos;
  latSpeed = P_Gain * _dPos_err;

  /****Turn****/
  _dTurn_err = _turnPos - desiredTurnPos;
  rotSpeed = P_Gain * _dTurn_err;

  /*Robot Output*/
  if (latSpeed > MaxSpeed)
  latSpeed = MaxSpeed;
  if (latSpeed < -MaxSpeed)
  latSpeed = -MaxSpeed;

  if (rotSpeed > MaxRotSpeed)
  rotSpeed = MaxRotSpeed;
  if (rotSpeed < -MaxRotSpeed)
  rotSpeed = -MaxRotSpeed;

  double _rightSpeed = latSpeed + rotSpeed;
  double _leftSpeed = latSpeed - rotSpeed;
  robot -> setVel2(_rightSpeed, _leftSpeed);
}
```

In real practice, a desired distance and a velocity limit is set for both lateral and rotational motion control. According to Equation (3), we leverage the rotational motion by applying a threshold value on turning dictation. If the turning distance is between the threshold value, as shown in Figure 4, the robot will identify the target as if it is in the middle of the sight, thus stabilizes the rotational motion of the robot, preventing it from oscillating indefinitely when the target is near the middle line. The output of rotational speed will be added or subtracted from the lateral speed, and we will get the right speed and left speed respectively. Eventually, PetBot will perform the speeds with the Aria function $robot- > setVel2$, which imposes the speeds on the right and left wheel.

2) Happy and Spinning: $Happy$ and $Spinning$ functions orchestrate a sequence of movements to convey a sense of happiness or react to the command from the owner.

```
void Happy(ArRobot *robot)
{
  robot -> setVel2(0,0);
  int time;
  std::cout << "Happy \n";
  robot -> setRotVel(800);
  ArUtil::sleep(time*1.2);
  robot -> setRotVel(-800);
  ArUtil::sleep(time*3.4);
  robot -> setRotVel(1000);
  ArUtil::sleep(time*10.5);
  robot -> setRotVel(0);
}

void Circle(ArRobot *robot)
{
  robot -> setVel2(0,0);
  ArUtil::sleep(20);
  /*Full circle = 100 * 3500*/
  robot -> setRotVel(100);
  ArUtil::sleep(3500);
  robot -> setRotVel(0);
}
```

IV. Results

A. Experimental Setup

The experimental trials were conducted in Yong-Ling Room 412, a spatial setting designed to replicate the ambiance of a residential living room. This environment closely emulates a home setting, thereby facilitating the emulation of scenarios wherein owners interact with the PetBot as an integral part of their daily routines.

The selected devices and equipment to facilitate our research are as follows:

- **RealSense D435i Camera:** Employed for capturing RGB-D information, the RealSense D435i boasts a remarkable frame rate of up to 90 fps. The RGB frame resolution is set at 1280 x 720, with an optimal depth range spanning from 0.3 to 3 meters. This camera seamlessly interfaces with computers via a USB3.1 port.
- **Lenovo ideapad 330S-14IKB Laptop:** The central computing hub for our experiments, the Lenovo ideapad 330S-14IKB, operates on the Windows 11 (64-bit) platform. Additionally, a virtual environment running Ubuntu 16.04 (64-bit) is hosted in Oracle VM VirtualBox to support specific aspects of our research.
- **Pioneer P3-DX Robot:** The Pioneer P3-DX is a compact two-wheel, two-motor differential drive robot designed for indoor laboratory and classroom applications. With a payload capacity of 17 kg, it comfortably accommodates the laptop and RealSense D435i camera. Connectivity to computers is facilitated through a USB2.0 port.

In this study, the experimental protocols comprise two primary programs: the Vision Program and the Robot Program. The Vision Program, executed in the Windows 11 environment and written in Python, is responsible for frame capturing, image processing, and computer vision tasks. On the other hand, the Robot Program are executed in the Ubuntu 16.04 environment and crafted in C++, operates the Pioneer robot, endowing it with the semblance of a responsive and intelligent pet. The intricacies of message exchange between these two programs are elaborated in III-C.

B. Implementation Details

In the experiments, we first tested the reaction time and reliability of the computer vision part, including distance detection, gesture recognition, and tennis ball localization. The results are illustrated in IV-C1, IV-C2, and IV-C3 respectively.

Secondly, we integrated the vision part and robot part and tested all the functions of PetBot, including searching, tracking, stopping, spinning around, being happy, and playing the tennis ball. We modified the parameters, such as the PID factors and max speed, to make the robot move smoother and mimic a real pet, and the results are shown in IV-C4.

The parameters we finally chose are listed in Table I. The turning threshold a discussed in Eqn. 3 is assigned to 80 when the image size (H, W) is (640, 480). As the distance estimation only relies on the measuring point located at the body in the image, the accuracy of the positions of individual body nodes does not need to be precise. The pose detection and tracking confidence is set to 0.5 to ensure sensitivity. The smooth factor α is set to be 0.4, which is obtained by trial and error to strike a balance betweeen the convergence and the smoothness.

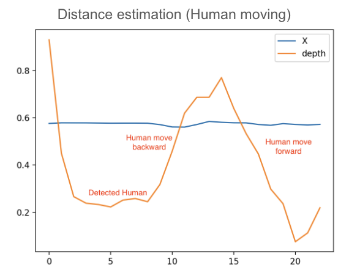

Fig. 5. Distance estimation of human moving backward and forward

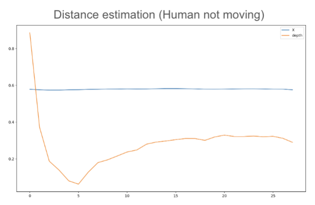

Fig. 6. Distance estimation of human not moving

TABLE I
PARAMETER SETTINGS

Parameters	Annotations	Values
Image width and height	*(H, W)*	(640, 480)
smooth factor	α	0.4
Turning threshold	a	80
Detection confidence	*conf*	0.5
Smoothing factor	*alpha*	0.6
Position P constants	POS_{KP}	1750
Position I constants	POS_{KI}	0
Position D constants	POS_{KD}	0
Turning P constants	$TURN_{KP}$	0.5
Turning I constants	$TURN_{KI}$	0
Turning D constants	$TURN_{KD}$	0

C. Experiment Results

1) Distance Estimation: The distance estimation experiments has been carried out several rimes. Initially, the distance estimation function is calculated by Mediapipe pose world landmarks, which claimed to return the value of depth distance in meters. Figure 5. and Figure 6. shows the beginning results of distance estimation. The blue line demonstrates the lateral movement, while the orange line shows the depth distance measurement. Since the person only walk backward and forward, there should be no lateral movement. The result has demonstrated good performance in lateral measurement. The depth measurement has a large error at the beginning. Despite the fact that the error can converge to the right value eventually, the measurement is too slow and noisy to be used in control program. Therefore, a modified version of

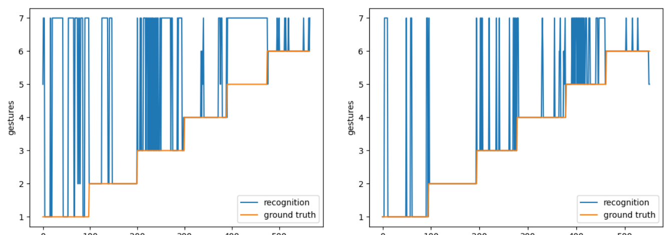

Fig. 7. The gesture recognition results without adjustment (left) and with adjustment (right). X coordinate is the number of frames, and Y coordinate is the gesture IDs. From 1 to 7 are 'happy', 'playing ball', 'stop', 'stop playing ball', 'tracking', 'spin around', and 'unknown', respectively.

distance estimation algorithm is proposed, which is mentioned previously. With the modified distance estimation method, the distance measurement is more reliable, which is validated in the control program.

2) Gesture Recognition: The precision of gesture recognition is subject to various factors, including illumination, background, hand-to-camera distance, and finger direction. The pre-trained MediaPipe package provided in cvzone allows us to minimize the impact of illumination and background conditions. MediaPipe exhibits optimal gesture recognition within a valid distance range of 0.5 to 3 meters which is sufficient for indoor applications. However, the accuracy of recognition is susceptible to finger direction, especially when employing the $fingerUp$ function. For example, extending all the fingers and showing the palms may not consistently register the thumb as "upwards," leading to potential gesture misinterpretation. To mitigate this, we adjusted gesture thresholds by incorporating additional conditions and the results are shown in Figure 7. Upon comparing the outcomes with the ground truth, it becomes evident that the recognition results following adjustment surpass those obtained without such refinement. This observation underscores the substantial improvement in accuracy and reliability achieved through the introduced adjustment in gesture recognition.

3) Tennis Ball Localization: In the context of tennis ball localization, challenges akin to those encountered in gesture recognition arise, such as illumination, background, and ball-to-camera distance. In an effort to mitigate the impact of illumination, our approach involves employing the HSV color space for thresholding instead of the RGB color space. This choice allows us to specifically target the desired color while minimizing the influence of illumination. Additionally, the rare occurrence of the tennis ball color in the background enables effective background removal by setting a sufficiently narrow threshold range. Moreover, the permissible ball-to-camera distance is within the range of 0.5 to 3 meters, constrained by the physical dimensions of the tennis ball.Consequently, the potential adverse effects of illumination, background interference, and ball-to-camera distance become negligible during the playing mode. However, limitations persist as the Petbot struggles to capture rapidly moving tennis balls, and its receiving angle imposes restrictions. Therefore, PetBot cannot act like a real dog to chase a ball when we throw it away. Nevertheless, the PetBot demonstrates optimal performance when tennis balls are moved or thrown at a sufficiently slow pace, allowing for smooth pursuit and interaction.

4) Integrated Function Experiment: This experiment is conducted to test the integration of a robot and a vision system, with a focus on feedback control as the primary testing object. As feedback control relies on signals acquired by the vision system as input, the relationship between the two systems is a crucial factor in determining the effectiveness of feedback control. The system integration primarily involves three components: the robot, vision, and communication. According to our tests, the communication system demonstrates the capability to transmit signals captured by the camera to the robot in real-time. Therefore, the experiment primarily focuses on observing the relationship between robot and vision system.

The impact of the vision system on the robot system is primarily manifested in the accuracy of signals and the reading frequency. The depth distance readings exhibit precise and high update-rate performance, while horizontal distance readings, although accurate, demonstrate only mediocre update rates. In the overall system presentation, the robot shows a good alignment with our expectations during forward and backward movements. However, issues arise during turns, where delays can occur, or when the owner exits the field of view, resulting in the robot failing to recognize and receive accurate position signals, causing it to rotate in the wrong direction.

To address this issue, without modifying the design of the vision system, we opt to equip the feedback control for rotation with a lower gain. This decision is made to prevent the robot from rotating abruptly, which could hinder the accurate reading of horizontal position signals. On the other hand, we discovered that the detection of a ball object is less sensitive than that of a human object. Thus, we optimize a gain with even lower value for the feedback control of ball tracking, which turned out to meet our needs.

V. Conclusions

This study introduces PetBot, a responsive companion robot aimed at alleviating loneliness among unmarried young individuals and addressing dementia in the elderly. PetBot utilizes real-time vision-based functions, responding to user gestures through tasks like tracking, stopping, displaying positive reactions, spinning, and playing tennis ball. It integrates gesture recognition and vision-based distance control, leveraging tools like OpenCV, MediaPipe, and cvzone for object detection and accurate distance estimation. The successful real-time communication between vision and motion programs demonstrates PetBot's integration and responsive control. Experimental results validate PetBot's proficiency in tasks, highlighting its potential to combat loneliness and enhance cognitive well-being. This work lays the groundwork for future pet-like companion robots.

WORK DIVISION

We divided our team into two groups: the Robotics Group and the Vision Group. The Vision Group, composed of YU-HAN TSAI and FENG-YI CHEN, focuses on image recognition. Our responsibilities include processing gestures, identifying the positions and distances of people and objects in the frame, and converting this information into commands and data. The Robotics Group, composed of FENG-EN YEH, SHIUAN-JEN YANG, and YU-HSIN CHEN, is primarily responsible for operating the Pioneer P3-DX. They use C++ to translate commands and data into the actual movements of the Pioneer robot.

REFERENCES

[1] Bradski, Gary. “The openCV library.” Dr. Dobb’s Journal: Software Tools for the Professional Programmer 25.11 (2000): 120-123.

[2] Lugaresi, Camillo, et al. “Mediapipe: A framework for building perception pipelines.” arXiv preprint arXiv:1906.08172 (2019).

[3] J. Liu, L. Li, Y. Liu and M. Faied, ”Autonomous Scouting Robot Based on Pioneer P3-DX,” 2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER), Suzhou, China, 2019, pp. 1392-1397.

[4] Panwar, Vikas Singh, et al. International Journal of Information Technology, vol. 13, no. 5, 18 July 2021, pp. 2101–2108.

[5] Apriaskar, Esa, et al. “Autonomous Mobile Robot Based on Behaviour-Based Robotic Using V-REP Simulator–Pioneer P3-DX Robot.” Jurnal Rekayasa Elektrika, jurnal.usk.ac.id/JRE/article/view/15081.

[6] Murtazahassan. “CV Zone.” https://github.com/cvzone/cvzone.