Implementation of Biped Robot with Smart Vision and Gait Controller

Kuo-Ho Su, *IEEE Member*, Chung-Hsien Kuo, *IEEE Member* and Che-Wei Hsu

***Abstract*—In this study, the deep learning technology is applied to recognize various objects and to reduce the computation burden of the biped robot. The YOLOV4 neural network was adopted to recognize the obstacles in the first part of this research. Because of YOLOV4's computational power, no need to connect to a server computer, the recognizing accuracy and fast computational characteristics, it satisfies the basic demand of smart biped robot. In order to achieve higher performance of obstacle avoidance, the second part of this research is to add the depth camera D435i which combines with the YOLOV4 neural network, the proposed architecture can more accurately measure the width and depth of the obstacles ahead. The third part of this research is to design a set of smart lightweight controller to reach above mentioned obstacle recognition ability. The fourth part is the mechanical design and gait control of biped robot. Some simulation and experimental results are provided in this paper. Furthermore, a biped robot prototype is implemented. We hope the implemented biped robot can not only replace our human beings to carry out the high risk jobs, but also can assist the transporting system.**

***Index Terms*—Biped robot, Gait control, Obstacle recognition, Raspberry Pi microcontroller, YOLOV4**

I. Introduction

Due to the development of Industry 4.0 [1], the industrial technologies have enabled us to work with machines in new intelligent ways. A new generation of automated humanoid robots is gradually being implemented. These robots, equipped with advanced controller, artificial intelligence (AI), sensors and machine vision, can execute difficult and complex tasks, recognize and analysis information from their surrounding environments.

This study is intended to compare the novel technologies of biped robots in terms of mechanism, balancing method and machine vision. First, the American Agility Robotic Company, released an ostrich biped robot Cassie[2]. The upper part of its legs is a six-bar mechanism and its purpose is to reduce the impact on the motor. The foot design method makes Cassie's agility and strength being better than human type. In addition, it's balancing method utilizes the hybrid dynamics with full model possessing 14 rotary encoders, IMU and camera. Another American robotics design company, Boston Dynamics, released the Atlas[3] biped humanoid robot. The Atlas controller is called the model predictive controller (MPC), which uses the robot dynamics model to predict robot's future movements. The Atlas has 28 hydraulically driven limbs and sensors such as IMU, camera and laser radar. ASIMO[4] is a humanoid robot developed by Honda Giken Industries from Japan. To balance the ground reaction force, ASIMO has to achieve the body posture control, which is called zero moment point (ZMP). When the ZMP control is activated, ASIMO can adjust the step size to restore the correct body position and velocity. The technology is called motion prediction control, also called iWalk. Using this technology, ASIMO can predict the center of mass during the walking actions and can predict the running time which is necessary to finish the action in real time.

After analyzing above technologies, the aim of this research is to implement another type of smart biped robot. We will focus on the utilization of Raspberry Pi 4B with the depth camera D435i and YOLOV4. The visual recognition function is used to calculate the length and width of the obstacles in front of the robot. Then the obstacles' coordinates can be obtained to determine whether the robot can move forward or not. Furthermore, the Linear Inverted Pendulum Model (LIPM) is adopted as the basis of balancing control. Summarily, the image recognition can be applied to identify the location and type of obstacles in front of the robot and the proposed recognition method combined with the biped robot can enable the robot to quickly avoid obstacles.

II. YOLOV4 Model TrainiNG Process And Results

This study will use the YOLOV4 [5] neural network and D435i depth camera as robot vision system. Some training process and results are described as follows.

A. YOLOV4 model training process

The built in Yolov4-custom.cfg is adopted as a template and some steps are modified as follows [6].
Step1: Modify the parameters of cfg files.
Step2: Use the labelImg (image labeling) tool to build the label position and to maintaining the integrity of the specifications as shown in Fig. 1.
Step3: Execute model training.

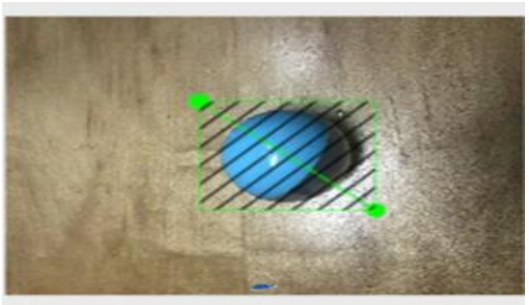

Fig. 1. Positioning the object

This research supported by the Ministry of Science and Technology of Taiwan, R.O.C. through its grant MOST 110-2221-E-034-015-.

Kuo-Ho Su is with the Graduate Institute of Digital Mechatronic Technology, Chinese Culture University, Taipei, Taiwan (e-mail: sgh@faculty.pccu.edu.tw)

Chung-Hsien Kuo is with the Department of Mechanical Engineering, National Taiwan University, Taipei, Taiwan (e-mail: chunghsien@ntu.edu.tw)

Che-Wei Hsu is with the Graduate Institute of Digital Mechatronic Technology, Chinese Culture University, Taipei, Taiwan(e-mail: dewey0824@gmail.com)

B. YOLOV4 training result

After the model training, the current loss rate and the number of training iterations are shown in Fig. 2. In this study, there are two categories of obstacles are trained. To training *YOLOV4*, 50 water bottles pictures and 50 boxes pictures are collected to train. The identifying results of the weights file for 2000 training times are shown in Fig. 3.

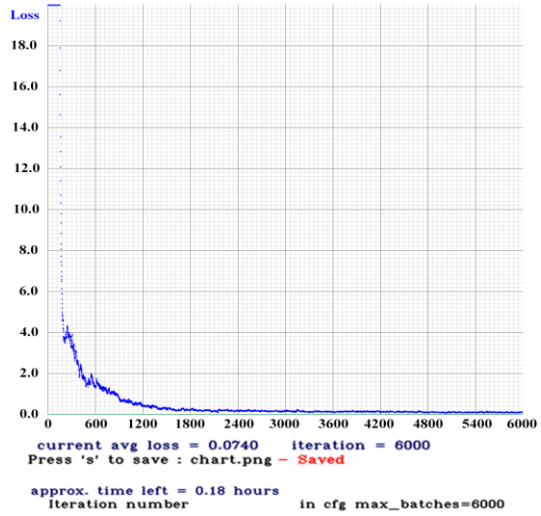

Fig. 2. Loss curve

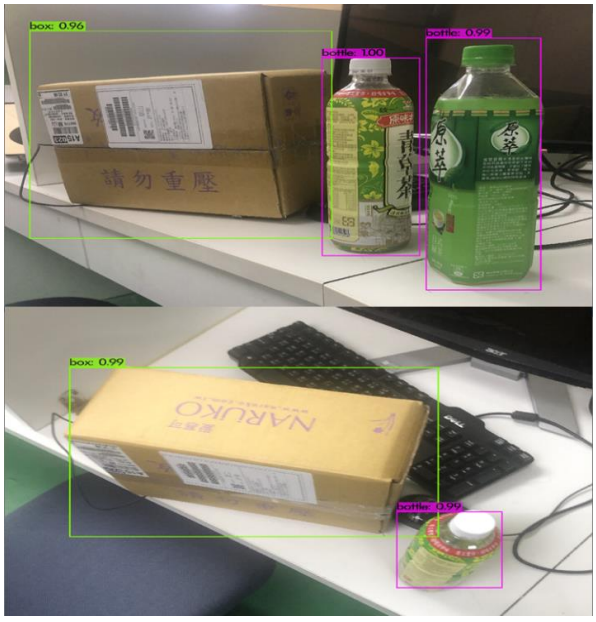

Fig. 3. Weights file identification results for 2000 training iterations

C. D435I depth camera software installation and use

This study adopts the RealSense SDK and the simplified installation steps are described as follows [7]:
Step1: Install the dependency package and download the RealSense SDK.
Step2: Start compiling after compiling preparation.
Step3: Set up udev rules to make RealSense-viewer recognize the device more easily.
Step4: Start RealSense-viewer and verify it.
Step5: Install the PYTHON interface library pyrealsense2.

III. Biped Mechanism And Smart Lightweight Controller Design

The Creo is adopted as the robot 3D modeling software tool during mechanism design stage [8]. LIPM is utilized as the robot balancing method and the Raspberry Pi 4B is used as the core controller. All the contents are described in the following subsections.

The designed biped robot in this research uses a half-body mechanism. To make the robot walking stably, the mechanism is developed by referring to the natural ostrich as the mechanism and adding a four-bar mechanism to increase the stability of the robot. The proposed mechanism 3D modeling is shown in Fig. 4.

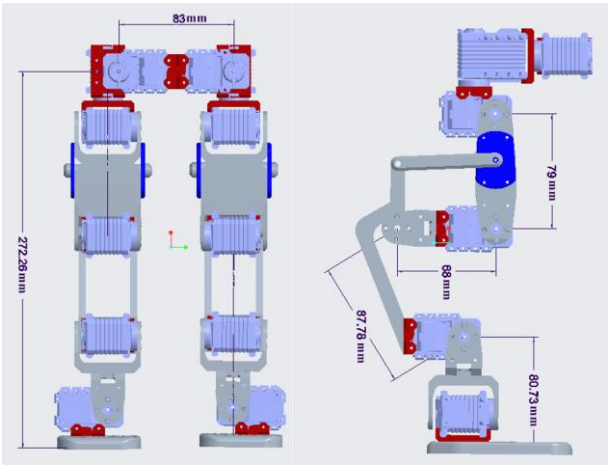

Fig. 4. 3D modeling of proposed biped robot

A. Two dimensional linear inverted pendulum (2D-LIP) balance control

[9] is referred first and then some simplified hypothesis is proposed as follows. The robot's whole body mass (m) is concentrated in the robot's center of mass position. The robot's legs are massless and have a scalable degree of freedom and intersect the ground only at one point, as shown in Fig. 5.

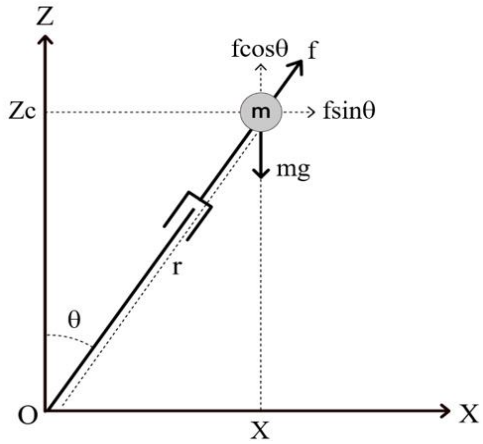

Fig. 5. Two dimensional linear inverted pendulum schematic

In 2D-LIP model, the center of mass must be maintained at the same height Zc when the center of mass will be far away from the support point and the support foot must be extended to maintain the center of mass at a fixed height. When the support

leg is extended, a force (f) is generated along the link between the support point and the center of mass. The force (f) can be split into f*cos(θ) as well as f*sin(θ), and mg is the force caused by the acceleration of gravity on the center of mass.

To keep the LIPM in continuous and stable moving, the landing point of each step is calculated by orbital energy, according to following equation.

$$\frac{1}{2}\dot{x}^2 + \frac{g}{2z_c}x^2 = E \tag{1}$$

Depending on the initial state of x (center of mass position) and ẋ (center of mass velocity), the LIPM motion can be divided into four oscillation cases, as shown in Fig. 6.

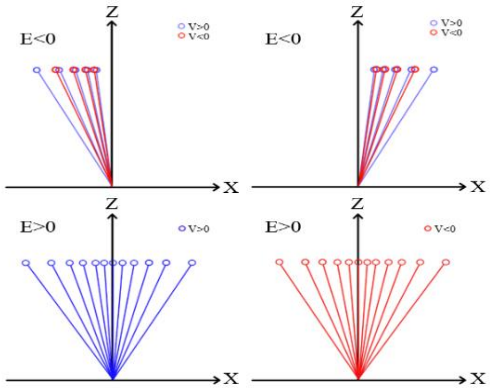

Fig.6. Four kinds of oscillations of 2D-LIPM

1) E<0: When the orbital energy is less than zero, the initial velocity is not fast enough. So the center of mass cannot cross the highest point of potential energy then center of mass number will drop to zero before the potential energy reaches the highest point and move in the opposite direction. Because of the different initial directions, there are two types of oscillations.
2) E>0: When the orbital energy is greater than zero. The center of mass crosses the highest point of potential energy and continues to move forward. Because of the different initial directions, there are two types of oscillations.

Capture Point [10] is calculated based on the current orbital energy E1 and the target orbital energy E2. Its purpose is to ensure that the next new orbital energy E1 is equal to the target orbital energy E2 when switching legs and to achieve this by controlling the step size. Fig. 7 shows the Python simulation results of the 2D-LIPM [11].

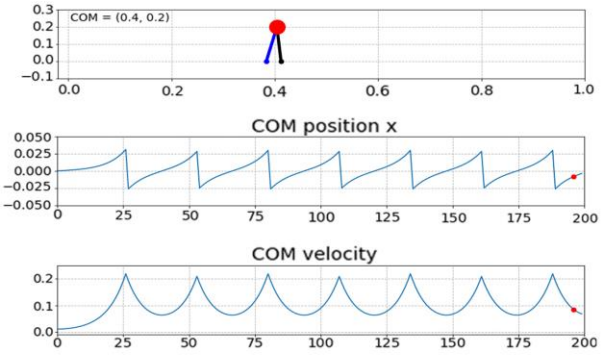

Fig. 7. 2D-LIPM simulation

B. Foot Track Curve

The desired foot track curves can be divided into X-axis and Z-axis and the equations are shown in following equations [12]:

$$x_S(t) = \frac{\frac{Length}{2\pi}\left[2\pi\frac{t-1}{T_S} - sin\left(2\pi\frac{t-1}{T_S}\right)\right]}{10}, 0 \le t \le T_S \tag{2}$$

$$z_s(t) = \begin{cases} \frac{\frac{Height}{2\pi}\left[2\pi\frac{t-1}{\rho T_S} - sin\left(2\pi\frac{t-1}{\rho T_S}\right)\right]}{10}, 0 \le t \le \rho T_S \\ z_s(\rho T_S) - \frac{\frac{Height}{2\pi}\left[2\pi\frac{t-1}{\rho T_S} - sin\left(2\pi\frac{t-1}{\rho T_S}\right)\right]}{10}, \rho T_S \le t \le T_S \end{cases} \tag{3}$$

where *Length* is the length of a step, *Height* is the maximum height of a step, ρ is the proportion of time from the step to the maximum height and T_S is the movement time of a step. Fig. 8 shows the simulated result of foot track curve using MATLAB.

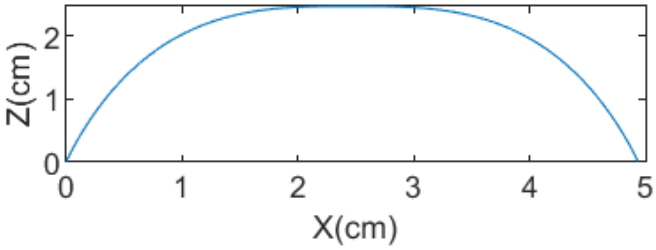

Fig. 8. Foot track curve

C. Inverse Kinematics (IK) and Forward kinematics (FK) simulation

In this study, we use the 2D triangular function to calculate IK by first removing the duplicate and unaffected mechanism then labeling the names on each nodes, as shown in Fig. 9.

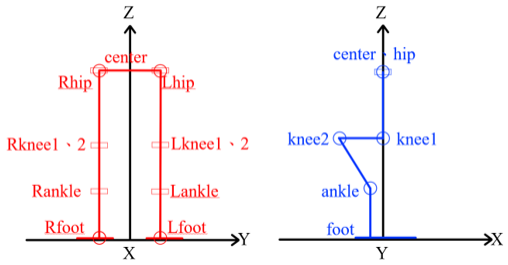

Fig. 9. Simplified foot mechanical structure chart

The trigonometric functions of IK derivation are described as follows. First bring in the end point and the center point, RFoot, LFoot and Center. Then the ankle coordinates are defined as

$$\text{RAnkle} = (\text{RFoot}_\text{X}, \text{RFoot}_\text{Z} + 80) \tag{4}$$

$$\text{LAnkle} = (\text{LFoot}_\text{X}, \text{LFoot}_\text{Z} + 80) \tag{5}$$

Furthermore, the hip coordinates are defined as

$$\text{RHip}_\text{X} = (Center_X, \text{Crnter}_\text{Z}) \tag{6}$$

$$\text{LHip}(Center_X, \text{Crnter}_\text{Z}) \tag{7}$$

As shown in Fig. 10, the proposed biped robot is a parallel four bar mechanism so the knee1 angles can be converted by the length (L), as shown in Eqs. (8) and (9).

$$\text{Knee1}=0.85(\text{L})-41.58 \tag{8}$$

$$\text{Knee2} = 50 - (90 - \text{Knee1}) \tag{9}$$

where L is the length of the structure and Knee2 can be calculated from Knee1 using the trigonometric function.

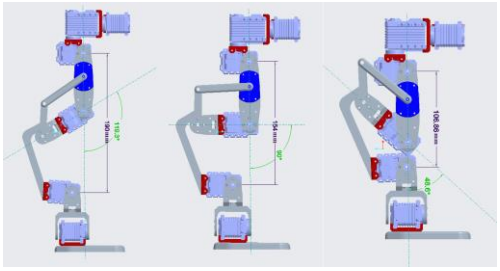

Fig. 10. Proposed four bar mechanism

$$\text{a} = \tan^{-1}\left(\frac{h}{w}\right) \tag{10}$$

where a is the Hip and Ankle related angle, schematically as shown in Fig. 11.

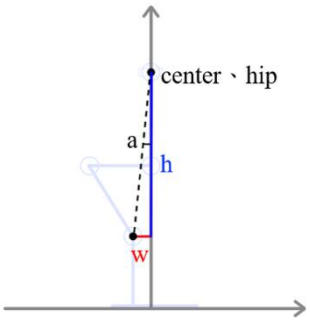

Fig. 11. Calculation of angle a

After getting the angle of each joint, substitute the angle into the FK and simulate to check whether it matches the foot track curve. The orange line of Fig. 12 is the actual foot curve of the FK simulation.

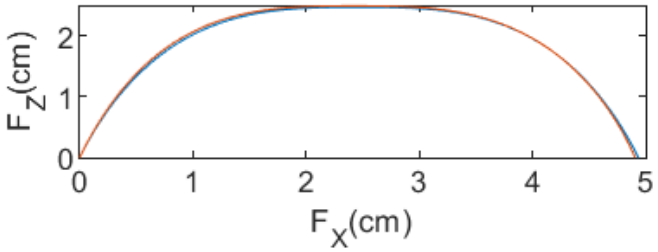

Fig. 12. Foot trajectory curve simulation results

D.Hardware Architecture

The proposed hardware architecture can be divided into four parts. The main part is the Raspberry Pi 4B. The second part is the D435i depth camera. The third part is the Arduino control board which is connected to Raspberry pi and receive its commands. The fourth part is the servos for each joint. The whole proposed hardware architecture is shown in Fig. 13.

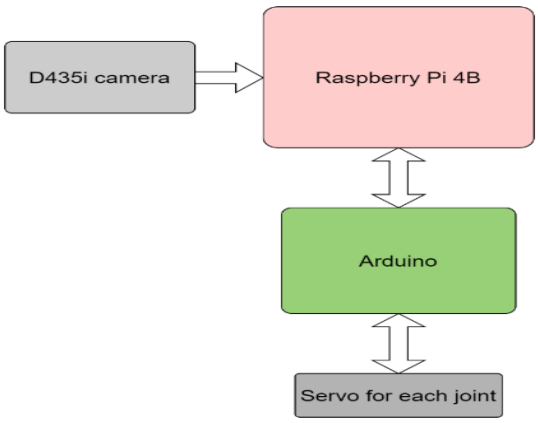

Fig. 13. Proposed hardware architecture

E. Software Architecture

The proposed software architecture can be divided into two parts, namely visual recognition and robot movement, as shown in Fig.14. The first part is used to train YOLOV4 on the computer side, four files are required as follows:

1. Model training parameters cfg file.
2. File path data file.
3. Training target names file.
4. Training target image jpg file.

After the training on the computer side, the weights file can be exported. Select the useful weights file and put it into the raspberry Pi and execute YOLOV4 recognition with the depth camera D435i.

The robot movement is firstly simulated by MATLAB on the computer, then the foot curves are entered and converted into the angles of each servo. Finally, the weights and rotation angles are embedded into Raspberry Pi and the final servo's commands are sent to Arduino after the operations of Raspberry Pi.

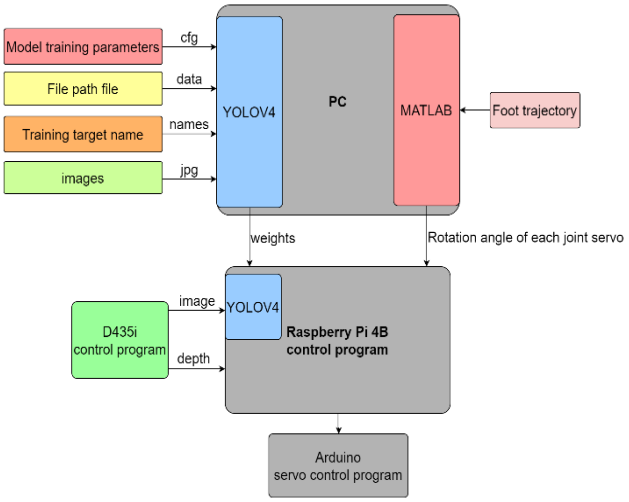

Fig. 14. Proposed software architecture

F. Program Flowchart

The program flowchart is divided into two major items, the orange part and the blue part as shown in Figs. 15-16. The orange part is the program flowchart of the Raspberry pi. From

left to right: raspberry Pi main control program, giving commands to the Arduino control board, going around the target, judging whether there is enough distance between two obstacles to pass, moving the robot parallel to the two obstacles, passing the two targets and moving sideways in clockwise and anti-clockwise. The blue part is the Arduino program flowchart. From left to right are: Arduino main control program and commands A to F. The order is: forward, left turn, right turn, left move and right move.

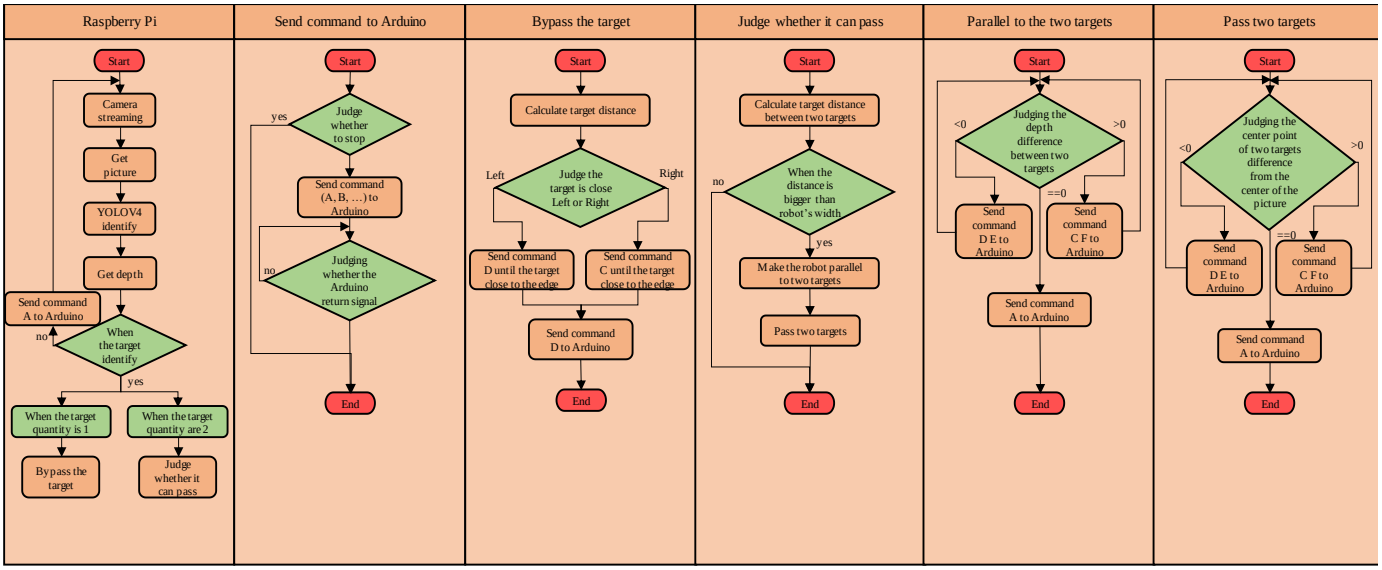

Fig. 15. Proposed program flowchart 1

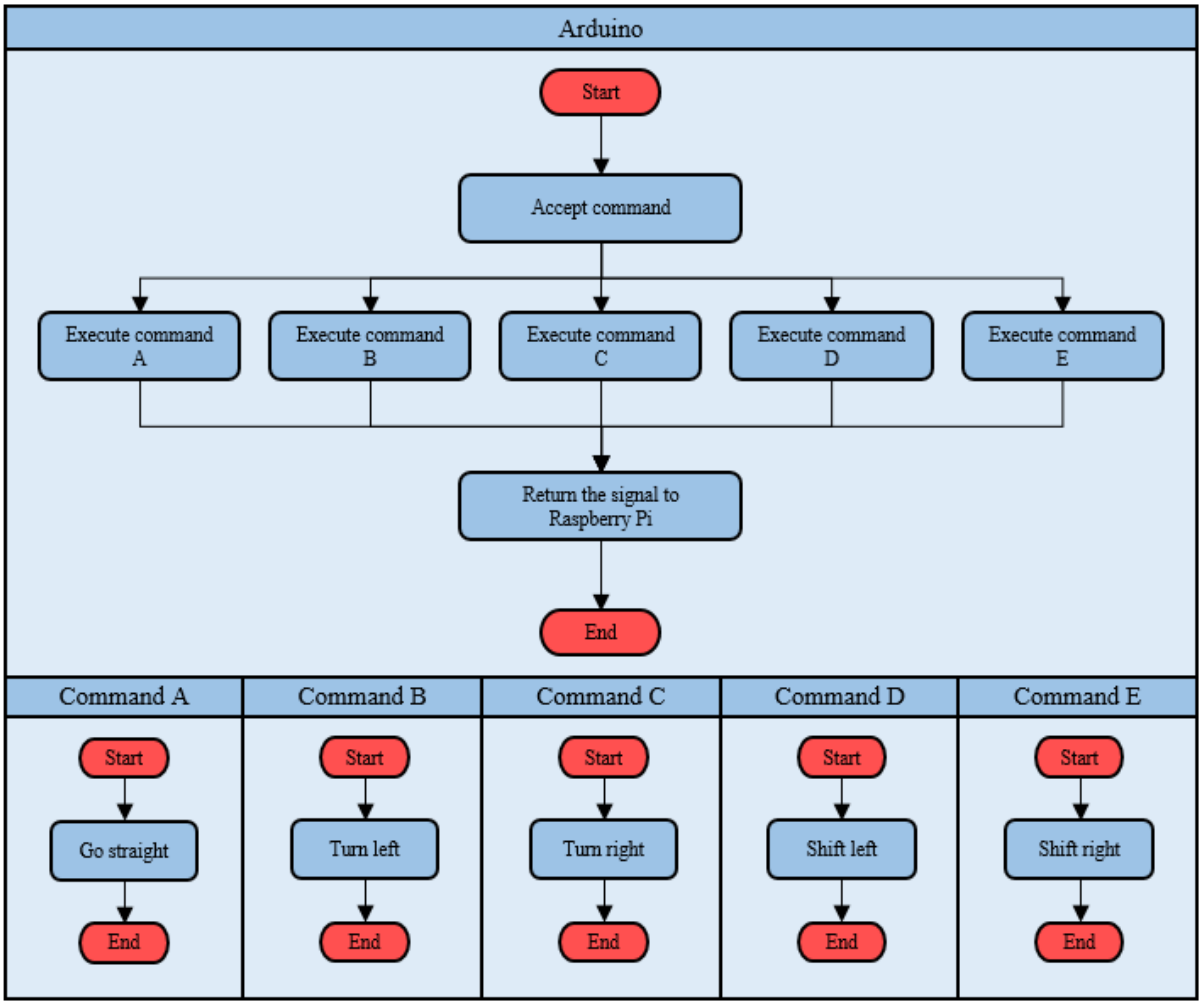

Fig. 16. Proposed program flowchart 2

G.LIPM Control Flowchart

After defining the omnidirectional walking instruction and gait parameters, we plan the LIPM and foot trajectory at the same time and bringing IK and converted into servo angles for each node [12] as shown in Fig. 17.

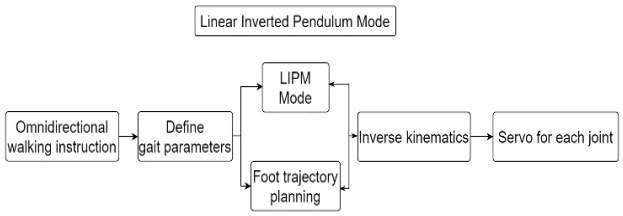

Fig. 17. LIPM control flowchart

IV. Implementation Results And Discussion

A. YOLOV4 Obstacle Identification and Measuring

In this subsection, some implementation results of identification are provided. The actual distance between the target objects is measured to determine whether robot can pass or not. In this study, we obtain the actual depth of the target objects by D435i [13] and compare the difference between the picture distance and the actual distance. Then convert it into the actual length between the two objects by trigonometric function. The results as shown in Fig. 18.

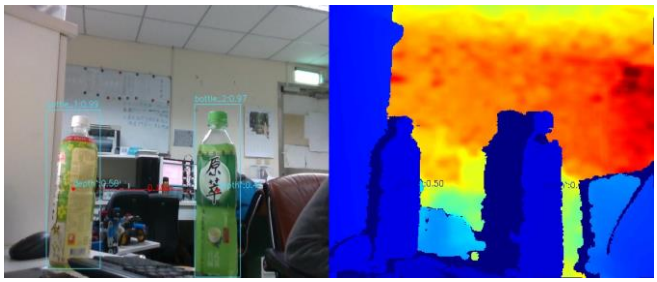

Fig. 18. Measured results of width and depth

B. Biped robot prototype and walking

In this subsection, some implementation results of prototype and walking are provided. To show the results, the walking process is divided into following stages and some disassembly graphs are shown in Figs. 19-23.

1. Squat (biped support): To increase the stability of the robot's center of gravity.
2. Move the center of gravity (biped support): Move the center of gravity to one of the feet, when the other foot step out can be stabilizing the body.
3. Step out of a foot (biped support > single foot support > biped support): The robot to stabilize the body with one foot, the other foot steps out in the stable foot's center of gravity.
4. Move the center of gravity (biped support): Move the center of gravity to the step out of foot, when the other foot can stabilize the body.
5. Step out of the other foot (biped support > single foot support > biped support): Continue to cross forward or close foot.
6. Move back to the body's center of gravity (bipedal support): The robot's center of gravity back to the central of feet.
7. Stand up (bipedal support): Back to the initial state.

The following five total movements: forward, turn left, turn right, left move, right move.

C. Discussion

According to above results, some comparisons and analyses are carried out in this subsection. The contents include hardware design, balance method and gait pattern planning.

In terms of hardware design, the human body as the entire structure is usually referred [14]. But in this paper, the bird structure needs fewer motors and is easier to achieve balance.

Many balancing methods have used reinforcement learning instead of manual adjustments. In [15], it is the ZMP-based fuzzy posture controller (FPC), which is a self-learning enhanced fuzzy posture controller. The FPC requires manual time to adjust the parameters, so it is combined with reinforcement learning to replace the manual adjustment of the fuzzy controller, so that the robot can learn itself. This paper uses the more basic linear inverted pendulum model as the balance method. In compare, the balance accuracy is lower, but the calculation quantity is smaller, so it can be used in small control boards.

In gait pattern planning, one of the methods [16] is to use the cycloidal profile and cubic spline interpolation to planning the robot's gait, and perform inverse kinematic calculations. In this paper, we used the foot trajectory planner. It can freely adjust the parameters of the foot trajectory to fit the linear inverted pendulum model and perform inverse kinematic calculations.

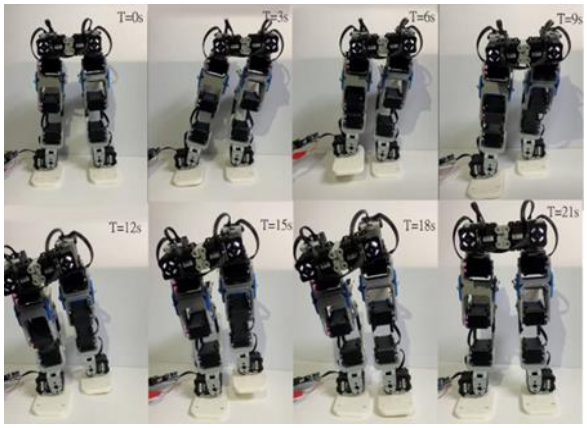

Fig. 19. Robot forward disassembly graphs

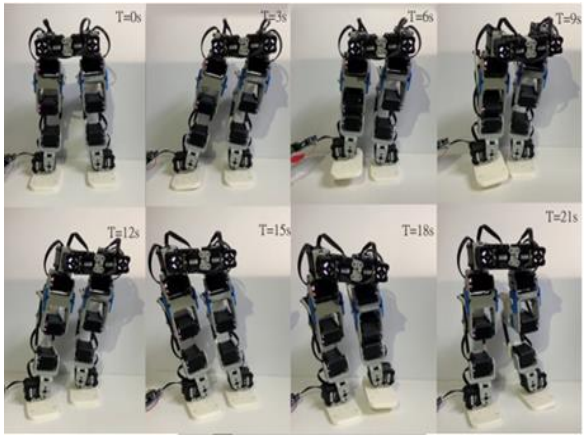

Fig. 20. Robot turn left disassembly graphs

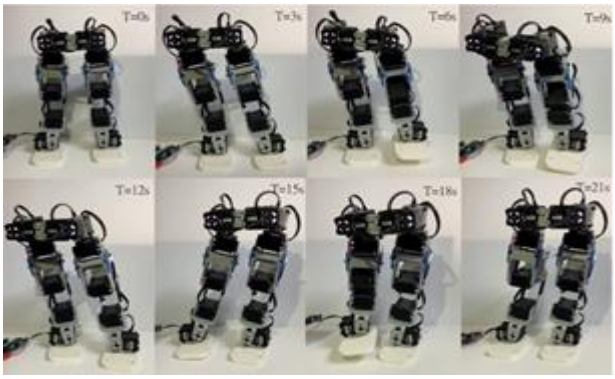

Fig. 21. Robot turn right disassembly graphs

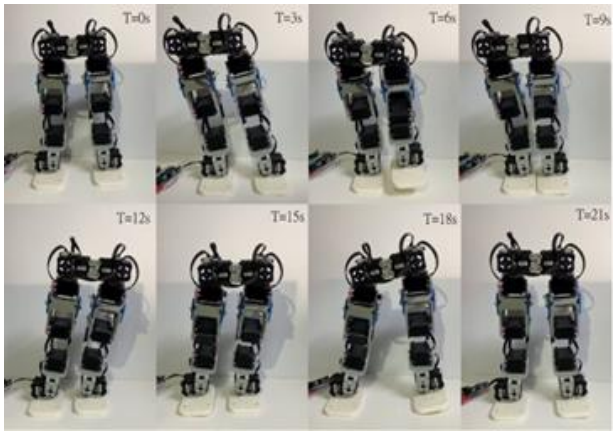

Fig. 22. Robot left shift disassembly graphs

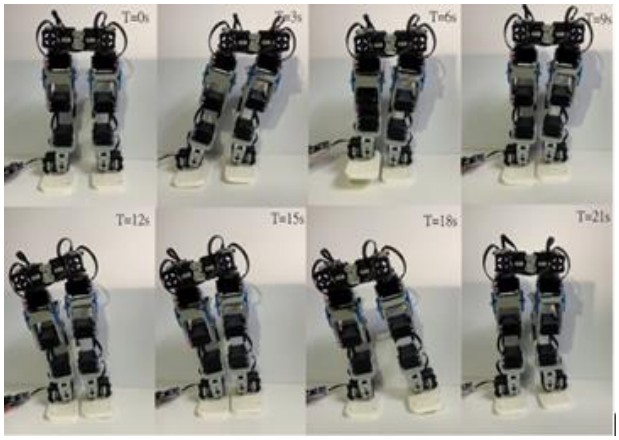

Fig. 23. Robot right shift disassembly graphs

V. Conclusion

This research examines how to use the visual identification to achieve obstacle avoidance. However, there are still some problems in balance, mobility and obstacle avoidance functions. We will improve it in the future.

(1) The power supply is necessary to power the controller and all the servos. Because of the large and heavy size of the power supply. It limits the movement of the robot so at least a 12V and 1.5A lithium battery is required to power it.

(2) The servo angles converted by IK need more fine tuning tests.

(3) To furtherly improve its balance, the 3D-LIPM and ZMP feedback control is necessary to solve this problem in the future.

(4) YOLOV4 still takes tens of seconds to identify when running in Raspberry Pi. This problem can be solving by using Neural Compute Stick2 to enhance the computational power or retraining a lighter YOLOV4 model to reduce the computational burden of Raspberry Pi.

(5) D435i depth camera with YOLOV4 recognition capability is limited. It need to train more targets and adjust the robot to recognize obstacles.

Acknowledgment

This work is financially supported by the Ministry of Science and Technology, Taiwan, R.O.C., under Grant MOST 110-2221-E-034-015-.

References

[1] What is Industry 4.0. [Online]. Available: https://www.sap.com/taiwan/insights/what-is-industry-4-0.html, Jan, 2021.

[2] J. Reher, W. L. Ma, Ames, A. D. Ames,"Dynamic walking with compliance on a cassie bipedal robot," In 2019 18th European Control Conference (ECC), IEEE. pp. 2589-2595, Jun, 2019.

[3] S. Feng, X. Xinjilefu, C. G. Atkeson, J. Kim, "Optimization based controller design and implementation for the atlas robot in the darpa robotics challenge finals," In 2015 IEEE-RAS 15th International Conference on Humanoid Robots, pp. 1029-1030, Nov, 2015.

[4] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, K. Fujimura,"The intelligent ASIMO: System overview and integration," In IEEE/RSJ international conference on intelligent robots and systems (Vol. 3, pp. 2478-2483), Sep, 2002.

[5] A. Bochkovskiy, C. Y. Wang, H. Y. M. Liao. "Yolov4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, pp. 1-13, 2020.

[6] Identify custom object with Labellmg. [Online]. Available: https://wings890109.pixnet.net/blog/post/68939643-yolo-v4, May, 2020.

[7] Linux Ubuntu Installation. [Online]. Available: https://github.com/IntelRealSense/librealsense/blob/master/doc/installation.md Oct, 2021.

[8] The Creo website. [Online]. Available: https://www.ptc.com/en/products/creo, Oct, 2021.

[9] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa. "Biped walking pattern generation by using preview control of zero-moment point," In 2003 IEEE International Conference on Robotics and Automation, Vol, No.2, 03, pp. 1620-1622, Sep, 2003.

[10] J. Pratt, J. Carff, S. Drakunov, A. Goswami, "Capture Point: A Step toward Humanoid Push Recovery" 2006 6th IEEE-RAS International Conference on Humanoid Robots, pp. 200-207, 2006.

[11] BipedalWalkingRobots. [Online]. Available: https://github.com/chauby/BipedalWalkingRobots, Jan, 2022.

[12] Q. Y. Lin, C. H. Kuo, S. H. Chiu, M. F. Li, Y. H. Fan, P. S. Lin, "Intelligent Robots - Principles and Applications," Publisher: Gau Lih Books, 01, pp.275-277, Dec, 2021.

[13] A. C. TSAI, "Discussion on the Feasibility of Depth Camera in Structural Monitoring," Master's Thesis of Civil Engineering and Disaster Prevention, Department of Civil Engineering, National Taipei University of Technology, Taipei City, pp.4-18, Jan, 2020.

[14] C. Y. CHEN, "Stabilizing Gravity of Biped Robot by PID Controller'z" [Online]. Available: https://hdl.handle.net/11296/t2bs2s, Jul, 2016.

[15] T. Y. CHEN, " The Study of Biped Robot Walking by using Reinforcement Learning and Fuzzy Logic Controller" [Online]. Available: https://hdl.handle.net/11296/55ryhf, Jul, 2019.

[16] C. C. Yung, "Gait Simulation of Biped Robot Based on ROS" [Online]. Available: https://hdl.handle.net/11296/khqk68, 2019.

Kuo-Ho Su was born in Yunlin, Taiwan, R.O.C., in 1959. He received the B.S., M.S., and Ph.D. degrees in Electrical Engineering from Tatung University, Taipei, Taiwan, R.O.C., in 1983, 1985, and 2005, respectively. Currently, he is a Professor with the Graduate Institute of Digital Mechatronic Technology, Chinese Culture University, Taipei, Taiwan, R.O.C. His research interests include the design and application of intelligent controller (fuzzy, neural network, genetic algorithm), robot system and embedded MCU system.

Chung-Hsien Kuo received the M.S. and Ph.D. degrees in Mechanical Engineering from National Taiwan University in 1995 and 1999, respectively. Currently, he is the Chairman and Professor of Department of Electrical Engineering, National Taiwan University of Science and Technology. He also is the Director of Taiwan TECH Industry 4.0 Center. His research areas include autonomous mobile robots, autonomous driving techniques, artificial intelligence, deep image learning, interactive cognitive learning, intelligent systems for Industry 4.0.

Che-Wei Hsu was born in Taipei, Taiwan, R.O.C., in 1998. He received the B.S. degree in Mechanical Engineering from Chinese Culture University in 2020. Currently, he is a master student with the Graduate Institute of Digital Mechatronic Technology, Chinese Culture University, Taipei, Taiwan, R.O.C.