2023 IEEE International Conference on Robotics and Biomimetics (ROBIO) | 979-8-3503-2570-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/ROBIO58561.2023.10354893

Visual Servo Control for Ball-on-Plate Balancing: Effect of PID Controller Gain on Tracking Performance

Sarin Kitchatr, Aphiphu Sirimangkalalo, and Ronnapee Chaichaowarat*, *Member, IEEE*

***Abstract*—Ball-on-plate balancing is a popular challenge for the implementation of control systems. In this paper, the Universal Robots UR3 industrial arm is set up for the challenge. An acrylic plate is firmly grasped by the robot's gripper. The pure rotation about three axes is achieved at the center of the plate by considering the offset of the plate from the tool frame. A web camera is mounted on the last link to observe the position of a ping pong ball moving on top of the plate. The color-based object detection is used with OpenCV and the coordinates of the ball are mapped to the pixel of the camera. At the beginning of each test, the ball was placed away from the center of the plate. Considering the x-axis and y-axis error of the ball position away from the center of the plate, the robot arm stabilizes the ball to the center according to the PID controller gain configurations. The effect of the proportional gain on reducing the rise time of the response is observed. The advantage of the derivative gain on reducing the response overshoot is observed. The advantage of the integral gain on reducing the steady-state error is not clearly observed for this ball-on-plate balancing problem. The implemented controller is robust against the small vibration of the camera with respect to the robot arm.**

I. Introduction

Serial manipulators, designed as a series of rigid links connected by joints that extend from a base to an end-effector, are one of the most common types of industrial robots and used in other applications e.g., laboratory liquid handling [1], ultrasonography training [2], and desktop arm rehabilitation [3]. The advances in physical human–robot interaction e.g., impedance control [4] [5], adjustable intrinsic properties of actuators [6] [7], soft sensors and actuators [8] [9], allow robots to collaborate safely with human counterparts in shared working environments [10] [11]. Image processing techniques capable of detecting human hand [12] and lower body limbs [13] make visual feedback more powerful for enhancing the capability of collaborative automation, and thus, allow human operators to pursue more valuable tasks and reduce their risk of injuries from repetitive motion and load carrying tasks.

The 2-DOF ball-on-plate balancing is a popular challenge for understanding the control system implementation. For the tracking control of a ball on beam, which is a nonlinear single-degree-of-freedom system, the approximate input-output linearization method was introduced [14]. The mechatronic design of a ball-on-plate balancing system was proposed [15]. The two motors were used for tilting the plate over the ball joint located beneath the plate's center via four-bar linkages. The resistive touch-sensitive glass screen (originally for a computer touchscreen) was used for sensing the ball position as feedback for a conventional PID controller. The similar setup was also implemented in [16]. The charge coupled device (CCD) camera was used to acquire the ball position and the fuzzy control scheme was introduced [17]. The linear quadratic optimal regulator (LQR) was implemented on the 2-DOF mechanical wrist system [18] for stabilizing the ball on plate by using a video camera to visualize the location of the ball in global frame. The 5-DOF manipulator was used to adjust the inclination of a beam rotating about a fixed pivot point (1-DOF) [19]. The RGB camera was used to observe (from the side view) the beam angle and the ball position error. The ROS-based software architecture was implemented on the 7-DOF manipulator [20] with the vision tracking system and the linear control technique was applied.

In literatures, the models were linearized around the center of the plate (or beam) although the axes of rotation were at the wrist joint of the manipulators [18] [20]. Even in the 2-DOF system specially designed for the ball-on-plate balancing [15], the center of the plate was located on top of the ball joint. The pure rotation (without translation) cannot be achieved at the ball height while located at the center of the plate. The position and orientation of the robot's end effector can be derived from the known joint displacements via the geometric model of the robotic arm [2]. However, the inverse kinematics for mapping the desired position and orientation of the end effector to the suitable joint displacements is a lot more challenging.

In this study, the Universal Robots UR3e collaborative robot allows achieving pure rotation about three axes at the center of the ball while located at the center of the plate by considering the offset of the plate away from the tool frame of the robot. A web camera is mounted on the last link to observe the position of a ping pong ball moving on top of the plate. The color-based object detection is used with OpenCV and the coordinates of the ball are mapped to the pixel of the camera. The effect of the proportion gain, derivative gain, and integral gain on the tracking performance is observed through the error response in both x-axis and y-axis.

This paper is organized as follows: Section II provides the setup overview of the UR arm, the camera mount, the python algorithm, and the visual tracking. Section III presents the ball-on-plate model and derives the transfer function of the control system. Section IV describes the experimental setup and discusses the results while section V summarizes the key findings.

II. System Overview

A. Universal Robot Manipulator

The industrial robotic manipulator UR3e was utilized as the motion platform for the simulations, providing the background for the real implementation. The UR3e features a

S. Kitchatr, A. Sirimangkalalo, and R. Chaichaowarat are with the International School of Engineering, Chulalongkorn University, Bangkok 10330, Thailand (e-mail: ronnapee.c@chula.ac.th). The first and second authors contributed equally to this work.

6-degree-of-freedom (6DoF) manipulator with a repeatability of 0.03 mm [21], ensuring precise positioning capabilities. This contrasts with other traditional 2-degree-of-freedom solutions or even the common 6DoF parallel manipulators [20] like the Stewart platform [22] while providing high adaptability and providing more movement in the robot space.

The UR3e, being a collaborative robot, offers the advantage of operating without the need for external protective barriers, enhancing safety and efficiency during implementation [23]. The absence of the need for an enclosing cage is particularly important for this application as externa human interaction with the ball is necessary [24]. Another benefit of the UR3e is its single arm design, simplifying the implementation for various tasks and extending its applicability to cooperative scenarios involving multiple manipulators.

B. Camera Mounting

As seen in Fig. 1, the camera will be mounted directly on top of an acrylic plate. The plate will be held by the gripper of the arm but can also be attached with an additional effector replacing the gripper. As the manipulator can hold up to 3 kg of payload [25], an aluminum profile is used to give the camera its required height to the plate to not waste 3d printing material. The 3d printed parts and the aluminum bar weight in total approximately 670 grams, which can be heavier for a stronger strength or lighter for smaller robotic arm. Acrylic plate is chosen for its relative smoothness and its lightweight properties and is held by the Hand-E gripper extension. The gripper limits the size of the plate as the gripper can only be extended to hold an object of approximate length of 13 cm.

C. Python Algorithm

Python, the open-source programming language known for its versatility and ease of use, has gained prominence as a powerful tool for software development [26]. In recent years, Python has been embraced within the robotics community for its applicability in creating integrated development environments (IDEs) and writing robot software. Control over the manipulator is done by connecting and sending data through TCP sockets which is connected by the robot IP through a LAN wire.

Similar to how ROS works with nodes, [20] the webcam sends data to the python computer which computes based on the ball position, the required angle of the robot arm to stabilize the ball. The angle is then sent to the robot as a pose in the form of $[x, y, z, r_x, r_y, r_z]$ which the robot controller will then calculate the kinematics to move the 6 joints to the required pose. This process is done continuously until the ball is in the set area. A simplified architecture is shown in Fig. 2.

D. Vision Tracking System Using Python

As stated before, a webcam is mounted directly on top of the plate to track the position of the ball. The object tracking process to obtain the position of the ball involves a series of image processing steps applied to the received image frames. This tracking methodology was previously employed in research conducted in [27], wherein an orange-colored object was effectively tracked using similar techniques. Using the function minEnclosingCircle() similar to [28], the ball

Fig. 1. Universal Robot UR3 and the camera mounting setup to enable visual servo control for Ball-on-Plate balancing. The pure rotation about three axes is achieved at the center of the plate by considering the offset of the plate from the tool frame.

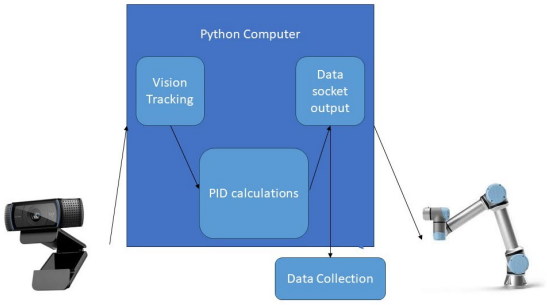

Fig. 2. Software architecture of the setup in python.

position coordinates can then be received which is then used for the calculations for the controller.

The low-cost Logitech C920 webcam is capable of producing image frames with a resolution of 1920×1080 pixels at a frame rate of 30 Hz. To facilitate accurate tracking, the camera is positioned on the fixture above the plate, covering an area of 13×20 cm². It is situated at a height of 40 mm from the plate's surface and is positioned 16 mm away from the supporting stand. This configuration allows the camera to capture the entire plate's surface, and if the gripper was removed, the maximum plate size can be increased to a 30x35 cm² with the current configuration.

III. Modeling of Control System

A. Ball on Plate Model

Various methods have been used to solve the ball and plate model such as using a resistive touch-sensitive glass plate [15] using phototransistors sensors [29] or various PID controllers: pole placement method [30], PID neural network [31], I-PD controller [32], approximate input-output feedback linearization for each decoupled ball and beam system [33].

For this study we worked using a PID controller equation in python to control the movement of the robot for ball stabilization. To make the system quickly respond to the error distance and external disturbance the Kp, Ki, and Kd values, which is the constant coefficient in the PID controller need to be tuned to make the system bring the ball back to the center of the plate as fast as possible. One method to tune the coefficient is the trial-and-error method with the simulation. Firstly, ball on plate system simulation is created in MATLAB. To accomplish the stated task, a dynamic model differential equation of the ball on plate system needs to be obtained. It should also be noted that the simulation is created to be applied only in one dimension or only for the ball on a slot model. This redundancy is minimal since the two axes of the ball on plate model follow the same dynamic model of the ball on a slot model. The mathematical model of the ball-plate balancing system is developed by using Lagrangian – Euler with the following assumptions taken from [34] – [36] had to be made for the mathematical model work correctly. First, the ball chosen is symmetric and smooth. Second, the ball does not slip. Third, the ball remains in contact with the plate surface all the time. Fourth, the friction in the system is neglected. Fifth, the minute translation movement of the system is neglected.

By assuming roll without slipping, the equation of motion relating the ball's linear acceleration along a single axis to the tilting angle of the plate was derived from the diagram shown in Fig. 3 [37].

$$\left(m_b + \frac{J_b}{r^2}\right)\ddot{x} + m_b g \alpha_x = 0 \tag{1}$$

where $\ddot{x}$ is the linear acceleration of the ball and α_x is the tilting angle of the plate. The values of parameters used for simulation are summarized in Table I.

The transfer function relating the ball's position to the tilting angle of the plate can be obtained by applying Laplace transform to (1), as follows:

$$\frac{X(S)}{A_x(S)} = \frac{m_b g r^2}{(m_b r^2 + J_b)S^2} \tag{2}$$

TABLE I. SYSTEM PARAMETERS USED FOR SIMULATION AND TESTING

System Parameters	*Variables*	*Values*	*Units*
Mass of ball	m_b	0.0027	kg
Radius of ball	r	0.02	m
Moment of inertia of ball	J_b	2.67×10^{-6}	kg·m^2
Gravitational constant	g	9.81	m/s^2
Width of plate	w	0.13	m
Length of plate	l	0.20	m

B. Simulation Results

With the transfer function and the values from the table at hand, the process of fine-tuning the PID controller gains is efficiently conducted using MATLAB. Using the pidtool() [38] function of the transfer function, the system parameters can be adjusted in the interactive screen shown in Fig. 4.

We set the maximum tilting angle of 7 deg for either side. By using the values from Table I, the time taken for the ideal rolling ball on an inclined surface to fall off at the longest

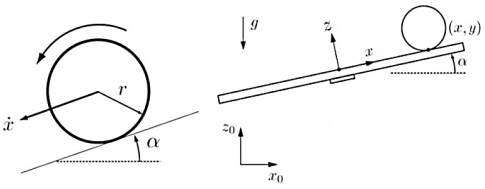

Fig. 3. Diagram showing model parameters of ball-on-plate system [37].

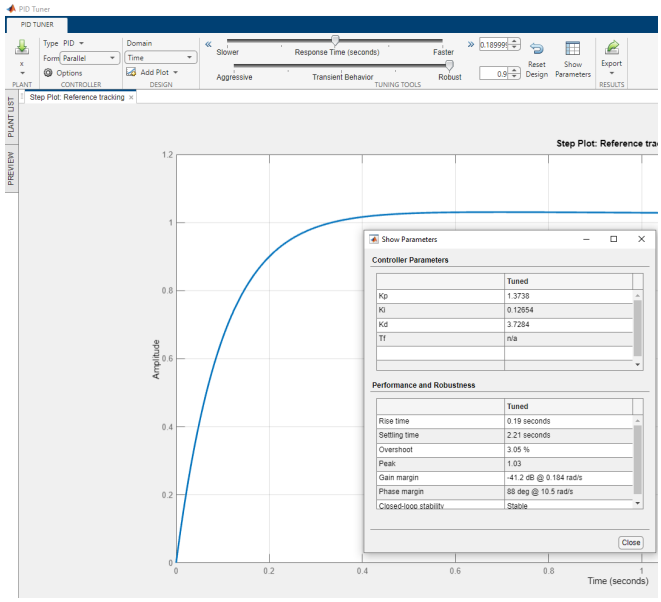

Fig. 4. Screenshot of MATLAB pidtool() window.

diagonal of the rectangular plate can be computed from (3). For the maximum tiling angle $\alpha_{max} = 7°$ the falling time $t \approx 0.28$ s.

$$t = \frac{7\sqrt{w^2 + l^2}}{5g\sin(\alpha_{max})} \tag{3}$$

Taking the rise time to be 67% of 0.28 seconds, 0.19 seconds, and that the system should be as robust as possible for any disturbance and the controller parameters can be exported as Kp=1.4, Kd=3.7 and Ki=0.13.

The controller parameters here will still need to be adjusted as these values are calculated on models with several assumptions but can be used as a base reference to begin tuning the controller.

IV. EXPERIMENTAL SETUP AND RESULTS

A. Experimental Setup

The webcam is connected to the computer through a simple USB wire, but with a sufficient speed can be connected wirelessly. The computer is connected to an OEM Control Box [25] provided by UR arm through a simple LAN wire at which the socket commands are sent. As the system is a one-way communication, the python code for receiving input from the robot arm was not given in the datasheet for the UR3e, the frequency of the systems is important to note. For safety reasons, any move commands sent to the robot takes 0.3 seconds to physically run. The robot can be forced to move in a shorter time, but due to how the robot stops after receiving a command, a shorter time will cause high vibration as the robot

is constantly stopping and then moving after each command. This difference in operating frequency also causes problems as the camera is receiving 30 frames per second, which can be considered equivalent to 30 Hertz, while the robot operates at a frequency of 3 Hertz, so an additional calculation is needed to ensure that the real time system is viable, without causing additional disturbances such as strong vibrations of the system. The diagram in Fig. 5 describes the experimental setup.

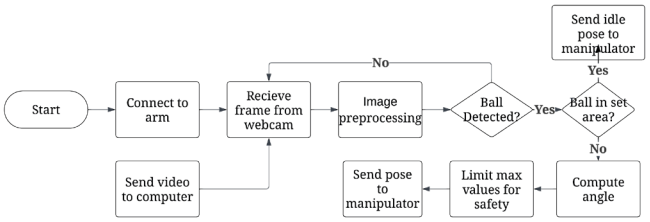

Fig. 5. Experimental process.

B. *Experimental Methods and Results*

With the values from the simulation results, we trialed heuristicially by testing values from 0 to 4 times the value found before, which means that Kp is trialed from 0 to 5.6, Kd is trialed from 0.0 to 14.8, and Ki is trialed from 0.0 to 0.52.

Firstly, we began trial and error for the Kd and Ki values for Kp=1.5. This is done by using the knowledge that Kd acts as a damper to account for the velocity of the ball, and Ki is introduced so that the system reaches the desired steady state response. The results are plotted in Fig. 6. An increase in Kp decreases rise time of the system to the predetermined setpoint which is expected from the change in Kp. A higher Kp than 5.3 only amplifies the ball movement causing the system to lose control and a lower Kp than 1.5 could not slow down the ball to a stable position during the experiment. The only exception is for Kp=5.3, which shows similar and slightly more unstable values to the effects where Kp=1.5. There is some amplification effects at Kp=5.3, but from 5 iterations of testing of each the 4 corners the ball remains stable when released from rest, and only when very small disturbances is added. When the ball is travelling from one end to another the angle changes from min to max adruptly which does in some case causes the ball to fall. For both X and Y, the system is overdamped as seen by the the oscillation under the set point. The significance in difference in set point is not only due to the steady state error but also a predetermined set offset which is used so that the robot arm does not do very minute changes.

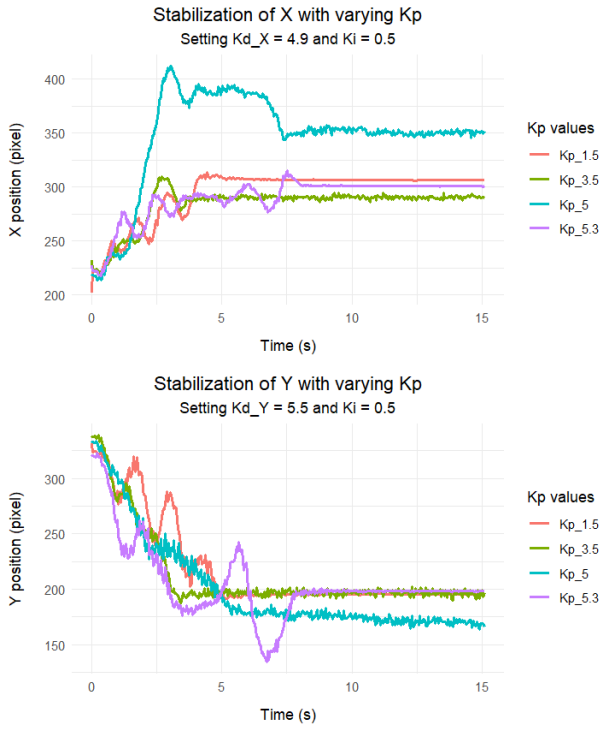

Fig. 6. Changes of x and y positions due to the values of Kp.

Through testing, a Kd less than 3.0 cannot stablize the oscillating pattern caused mainly by the proportional controller, and a values higher than 7.0 introduces another feeback loop which intensifies the input gain. Increasing Kd from 4 to 5 greatly reduces the overshooting effect and smooths the approach of the ball from its position to the set point. There is a slight consistent wobble at 1.5s for both x and y values, which we attributed to the static friction causing the values to spike up as more more gravitation force is required to move the ball i.e., increasing the angle to move the ball faster than the what is estimated from the formula.

As our system has some vibrations and is intended to be able to adapt to human disturbance, the effects of the Integral controller is needed. Another reason for adding an integrator is to reduce the system error; however, for this system the effect of the integrator is minimal. As a regular webcam is used for detection, the position resolution is not significant enough to produce a highly accurate result, an offset is used to reduce small minute changes when the ball is close to the setpoint, this is also why for each of the graphs in Figs. 6, 7 and 8, the graph does not end near to the setpoint. The X and Y error graphs also do not show a clear effect to the final result with varying Ki.

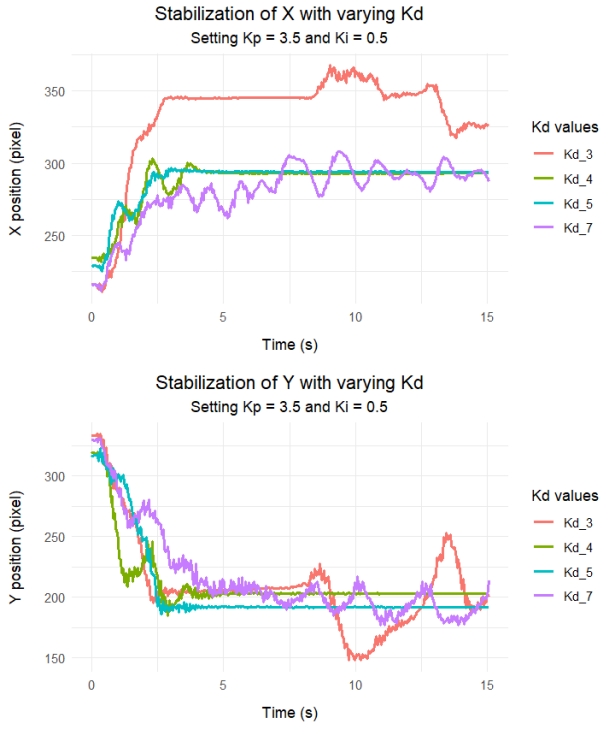

Fig. 7. Changes of x and y positions due to the values of Kd.

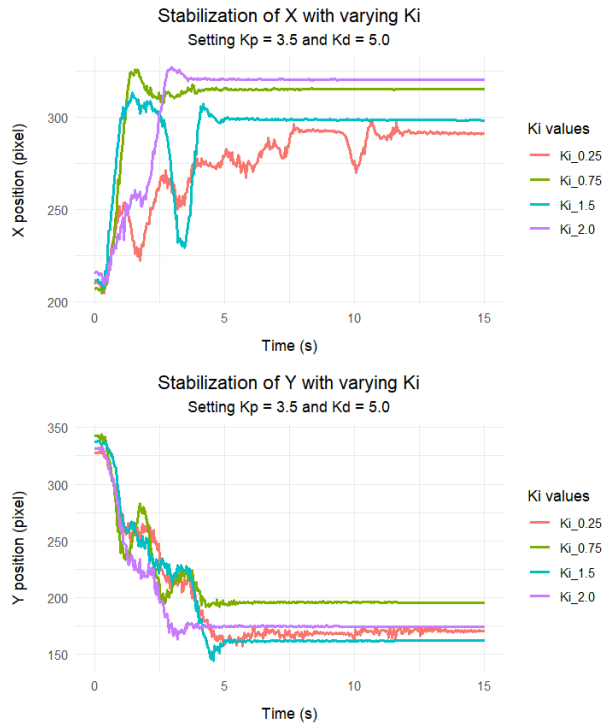

Fig. 8. Changes of x and y positions due to the values of Ki.

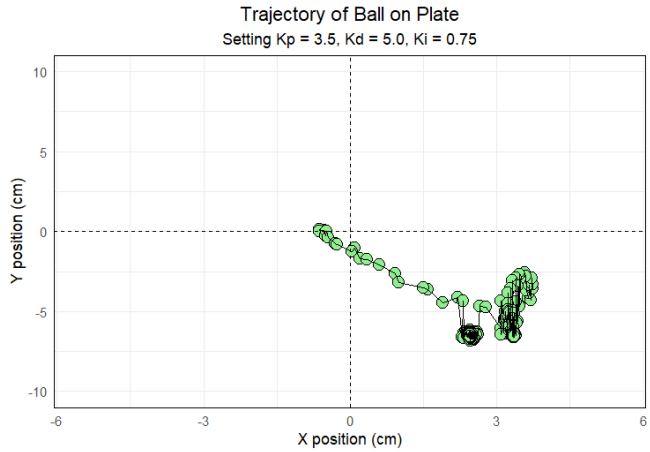

Fig. 9. Trajectory of the ball on the entire plate.

V. Conclusion

This study introduces a Python-based approach for controlling a 6-degree-of-freedom (6-DOF) manipulator arm that has been integrated with a vision tracking system. The design ensures seamless interaction between components, allowing the vision system and the UR3e arm to exchange state feedback and coordinate the computation and transmission of control commands for predefined control tasks. To illustrate its capabilities, the system was employed in simulating and conducting an experiment involving the stabilization of a ball on a plate. The experimental outcomes were successful, with the system effectively maintaining the ball's balance within a confined region near the center of the plate. The system also remains stable for various values of Kp Kd and Ki. The proposed system, along with the presented methodologies, holds potential for addressing more intricate tasks that involve a higher number of joint controls. Additionally, it can serve as an educational platform for students seeking to gain proficiency in both Python programming and control theories within the context of robotics.

Acknowledgment

This research project is partly supported by Tronormos Co., Ltd., Bangkok, Thailand.

References

[1] R. Chaichaowarat, A. Sirichatchaikul, W. Iamkaew, and N. Phondee, "Affordable pipetting robot: gripper design for automatic changing of micropipette and liquid volume control," in *Proc. IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics*, pp. 1275–1280, 2022.

[2] T. Mesatien, R. Suksawasdi Na Ayuthaya, A. Chenviteesook, and R. Chaichaowarat, "Position accuracy of a 6-DOF passive robotic arm for ultrasonography training," in *Proc. IEEE Region 10 Conf.*, pp. 841–846, 2023.

[3] T. Tantagunninat, N. Wongkaewcharoen, K. Pornpipatsakul, R. Chuengpichanwanich, and R. Chaichaowarat, "Modulation of joint stiffness for controlling the cartesian stiffness of a 2-DOF planar robotic arm for rehabilitation," in *Proc. IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics*, pp. 598–603, 2023.

[4] S. Nishimura, R. Chaichaowarat, and H. I. Krebs, "Human-robot interaction: controller design and stability," in *Proc. IEEE RAS/EMBS Int. Conf. Biomedical Robotics and Biomechatronics*, pp. 1096–1101, 2020.

[5] A. Javadi and R. Chaichaowarat, "Position and stiffness control of an antagonistic variable stiffness actuator with input delay using super-twisting sliding mode control," *Nonlinear Dyn.*, vol. 111, pp. 5359–5381, 2023.

[6] Z. Ullah, R. Chaichaowarat, and W. Wannasuphoprasit, "Variable damping actuator using an electromagnetic brake for impedance modulation in physical human–robot interaction," *Robotics*, vol. 12, no. 3, 80, 2023.

[7] R. Chaichaowarat, S. Nishimura, and H. I. Krebs, "Design and modeling of a variable-stiffness spring mechanism for impedance modulation in physical human–robot interaction," in *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 7052–7057, 2021.

[8] R. Chaichaowarat, S. Nishimura, T. Nozaki, and H. I. Krebs, "Work in the time of Covid-19: actuators and sensors for rehabilitation robotics," *IEEJ J. Ind. Appl.*, vol. 11, no. 2, pp. 1–10, 2021.

[9] K. K. Wichiramala, S. Opasjirawiroj, N. Chongpita, and R. Chaichaowarat, "Soft pneumatic actuator from 3D-printed TPU: fabrication and grasping force characterization," in *Proc. IEEE Region 10 Conf.*, pp. 853–858, 2023.

[10] R. Chaichaowarat, S. Prakthong, and S. Thitipankul, "Transformable wheelchair–exoskeleton hybrid robot for assisting human locomotion," *Robotics*, vol. 12, no. 1, 16, 2023.

[11] A. M. Abdullahi and R. Chaichaowarat, "Sensorless estimation of human joint torque for robust tracking control of lower-limb exoskeleton assistive gait rehabilitation," *J. Sens. Actuator Netw.*, vol. 12, no. 4, 53, 2023.

[12] K. Pornpipatsakul, A. Chenviteesook, and R. Chaichaowarat, "Ultrasound probe movement analysis using depth camera with compact handle design for probe contact force measurement," in *Proc. IEEE/EMBS Annu. Int. Conf.*, 2023.

[13] K. Pornpipatsakul, W. Chuengwutigool, R. Chaichaowarat, and A. Foongchomcheay, "Bridging exercise monitoring system using RGB camera for stroke rehabilitation," in *Proc. IEEE Region 10 Conf.*, pp. 959–964, 2023.
[14] J. Hauser, S. Sastry, and P. Kokotovic, "Nonlinear control via approximate input-output linearization: the ball and beam example," *IEEE Trans. Autom. Control*, vol. 37, no. 3, pp. 392–398, 1992
[15] S. Awtar, et al., "Mechatronics design of a ball-on-plate balancing system," *Mechatronics*, vol. 12, no. 2, pp. 217–228, 2002.
[16] M. M. Kopichev, A. V. Putov, and A. N. Pashenko, "Ball on the plate balancing control system," *Mater. Sci. Eng.*, vol. 638, no. 012004, 2019.
[17] X. Fan, N. Zhang, and S. Teng, "Trajectory planning and tracking of ball and plate system using hierarchical fuzzy control scheme," *Fuzzy Sets Syst.*, vol. 144, no. 2, pp. 297–312, 2004.
[18] C.-C. Cheng and C.-H. Tsai, "Visual servo control for balancing a ball-plate system," *Int. J. Mech. Eng. Robot. Res.*, vol. 5, no. 1, pp. 28–32, 2016.
[19] C.-L. Shih, J.-H. Hsu, and C.-J. Chang, "Visual feedback balance control of a robot manipulator and ball-beam system," *J. Comput. Commun.*, vol. 5, no. 9, pp. 8–18, 2017.
[20] K. A. Khan, R. R. Konda, and J.-C. Ryu, "ROS-based control for a robot manipulator with a demonstration of the ball-on-plate task," *Adv. Robot. Res.*, vol. 2, no. 2, pp. 113–127, 2018.
[21] Universal Robots. "UR3e technical details", Available: https://www.universal-robots.com/media/1802780/ur3e-32528_ur_technical_details_.pdf [Accessed: 15-Jul-2023].
[22] A. Kassem, H. Haddad, and C. Albitar, "Comparison between different methods of control of ball and plate system with 6DOF Stewart platform," *IFAC-PapersOnLine*, vol. 48, no. 11, pp. 47–52, 2015.
[23] R. Bogue, "Europe continues to lead the way in the collaborative robot business", *Ind. Robot*, vol. 43, no. 1, pp. 6–11, 2016.
[24] J. Fryman and B. Matthias, "Safety of industrial robots: from conventional to collaborative applications," in *Proc. German Conf. Robotics*, pp. 1–5, 2012.
[25] Universal Robots. "OEM Control Box Technical Sheet", Available: https://www.universal-robots.com/media/1808973/oem-control-box_tech-sheet_dec-2019.pdf [Accessed:15-Jul-2023].
[26] G. V. Rossum and F. L. Drake, *Python 3 Reference Manual*, CreateSpace, 2009.
[27] D. L. Baggio et al., *Mastering OpenCV with Practical Computer Vision Projects*, Packt Publishing, 2012.
[28] E. Ali and N. Aphiratsakun, "AU ball on plate balancing robot," in *IEEE Int. Conf. Robotics and Biomimetics*, pp. 2031–2034, 2015.
[29] A. Zeeshan, N. Nauman, and M. Jawad Khan, "Design, control and implementation of a ball on plate balancing system," in *Proc. Int. Bhurban Conf. Applied Sciences and Technology*, pp. 22–26, 2012.
[30] A. Knuplez, A. Chowdhury, and R. Svecko, "Modeling and control design for the ball and plate system," in *Proc. IEEE Int. Conf. Industrial Technology*, pp. 1064–1067, 2003.
[31] Z. Fei, Q. Xiaolong, L. Xiaoli, and W. Shangjun, "Modeling and PID neural network research for the ball and plate system," in *Proc. Int. Conf. Electronics, Communications and Control*, pp. 331–334, 2011.
[32] S. Mochizuki and H. Ichihara, "I-PD controller design based on generalized KYP lemma for ball and plate system," in *Proc. European Control Conference*, pp. 2855–2860, 2013.
[33] M.-T. Ho, Y. Rizal, and L.-M. Chu, "Visual servoing tracking control of a ball and plate system: design, implementation and experimental validation," *Int. J. Adv. Robotic Syst.*, vol. 10, no. 287, 2013.
[34] L. Spacek, V. Bobal, and J. Vojtesek, "LQ digital control of ball and plate system," in *Proc. Eur. Conf. Modelling and Simulation*, pp. 427−432, 2017.
[35] L. Spacek, J. Vojtesek, F. Gazdos, and T. Kadavy, "Ball and plate model for robotic system," in Proc. *Eur. Conf. Modelling and Simulation*, pp. 226–231, 2018.
[36] G. Madhumitha, S. Suresh Kumar, and A. Gurupriya, "Design and development of a ball-plate balancing system with a smart phone human-machine interface," *J. Phys.: Conf. Ser.*, vol. 1969, no. 1, 2021.
[37] M. Nokhbeh, D. Khashabi, and H. A. Talebi, *Modelling and control of ball-plate system*, Amirkabir University of Technology, 2011.
[38] MathWorks. "MATLAB, Control Toolbox, and Simulink", Available: https://www.mathworks.com/products/control.html [Accessed: 16-Jul-2023].