Development of a Control and Estimation Strategy for a Six-Link Robot Manipulator

Conference Paper · June 2020

DOI: 10.32393/csme.2020.1209

3 authors, including:

Stephen Andrew Gadsden
University of Guelph
149 PUBLICATIONS **2,733** CITATIONS

SEE PROFILE

Proceedings of the Canadian Society for Mechanical Engineering International Congress 2020
CSME Congress 2020
June 21-24, 2020, Charlottetown, PE, Canada

Development of a Control and Estimation Strategy for a Six-Link Robot Manipulator

Grant Hall[1], Andrew Lee[1], and S. Andrew Gadsden[1]
[1]College of Engineering and Physical Sciences
University of Guelph, Guelph, ON, Canada
Emails: hallg, alee32, and gadsden@uoguelph.ca

Abstract— This paper covers the ongoing approach of creating new combinations of robot controllers with filtering techniques. It outlines the need for such a system and a current controller system that uses filtering techniques to improve accuracy. It also explains how the proposed approach differs from the systems in literature. This paper outlines how the system is designed and tested. It explains the approach taken to design a system, and how the performance of the system will be measured. Finally, it explains progress that has been achieved, as well as recommendations for future work.

Keywords-Unscented Kalman Filter; Smooth Variable Structure Filter; DH Table; PID;

I. INTRODUCTION

Industrial robots have become an integral part in today's manufacturing landscape. They span a massive range of applications, from welding, grinding, packaging, shipping, or material handling. In all of these scenarios, it is important for the robots to have repeatability. Repeatability refers to the robot's ability to move to a specified pose, and then be able to return to that exact same position. Noise and disturbances caused by sensors or the environment can seriously impact a robot's accuracy, so often a closed loop feedback system is implemented to maintain accuracy [1, 2]. However, even with a closed-loop feedback system, these disturbances can still affect accuracy. One solution is to use filtering techniques to try and eliminate as much noise as possible. This short paper will discuss one of the possible controller/filter combinations that can improve accuracy as well as show steps taken to try and implement a PID controller and filter technique.

There have been many different approaches taken to improve accuracy or robot positioning. One approach by M. Cui, H. Liu, W. Liu, and Y. Qin used an unscented Kalman filter to attempt to estimate slip parameters of a wheeled robot [3]. The filter was combined with a unified controller with promising results. It was shown that the closed-loop system resulted in the tracking error to converge to zero, even with unknown slip values.

Another approach by K. Joni, M. Ulum, T. Prasetyo, and A. Y. Maulana focused on the optimal tuning of a PID controller and did not implement any versions of Kalman filtering [4]. The goal of the experiment was to develop a self-balancing robot with the ability to walk and run. Without the use of a filter, the tuning of the PID had to compensate for any noise in the system. While they were able to achieve steady-state PID values for stationary and walking conditions, there were unable to get steady-state PID values due to the quickly changing dynamics of the robot.

A third approach used a standard Kalman filter to assist in object tracking [5]. A mobile robot was equipped with a Microsoft Kinect camera to identify a candidate object, and Kalman filtering was used to help correctly estimate the relative distance to the object for the robot trajectory. B. Prasetio published a paper on using an Ensemble Kalman filter and PID controller on a self-balancing robot [6]. This paper showed very good robustness to system and sensor noise but required manual tuning since the noise covariance matrices were not known.

II. ESTIMATION STRATEGY AND PROPOSED CONTROLLER

The solution to increasing accuracy proposed in this paper is to develop a robot controller that will work in conjunction with different estimation strategies. The strategy uses an interactive multiple model (IMM) and an unscented Kalman Filter (UKF) and a smooth variable structure filter (SVSF) [7, 8, 9]. The UKF will help to accurately estimate the real world pose of the robot under normal operating conditions, and the SVSF will help predict the robot pose during non-standard conditions such as a faulty motor or an unexpected external force.

In order to achieve this, first a theoretical robot must be simulated. A virtual robot with characteristics based on the simplified geometry of an ABB 4600 industrial robot was implemented. These characteristics will include the DH table, link masses, link centers of mass, moments of inertia, and so forth. A simplified configuration was used in order to reduce computational load. This simulation will then give the robot a series of waypoints it must pass though, use inverse kinematics to solve the required poses, calculate a smooth trajectory using a spline function, and finally instruct the robot to complete its predetermined motion. As the robot moves through the simulation the position, velocity, and acceleration of each joint is recorded. These will be in the inputs to the new controller.

The proposed controller must be versatile in nature, and therefore will be designed in a way to allow universal applications [10]. This means that based on the physical characteristics of the robot, the controller must be able to reconfigure itself in order to handle any combination and orientation of revolute or prismatic joints, in a robot manipulator ranging from 1 to 6 degrees of freedom in 2 or 3-dimensional space. Using the characteristics provided, the first step is to initialize the robot workspace. The DH table used for the simulations can be seen below in Table 1.

TABLE I. DH TABLE USED IN SIMULATION

$\boldsymbol{a}$	$\boldsymbol{\alpha}$	$\boldsymbol{d}$	$\boldsymbol{\theta}$
0	0	0	θ_1
0.5	-90	0	θ_2
0.5	0	0	θ_3

III. INDUSTRIAL ROBOT SYSTEM

In order for the controller to be able to control the robot, the equations of motion for each link must be calculated. The Lagrangian method was used in order to calculate these equations. The key components of the equations of motion are the link mass, the torque applied to the link, and the angle of the link relative to the previous link. For simplicity, the mass of each link was taken as a point mass at the center of each respective link. The Lagrangian method of calculating equations of motion uses the principles of total kinetic and potential energy of the system, with the energy of each previous link contributing to the links that come after it.

The first step in calculating the Lagrangian method is to define the positions and velocity of the link masses. The first step in calculating the location of each link's point mass is to obtain a relationship between the known reference frame of the robot and each link. This is done using transforms matrices which represent rotation and translation about each axis. The transform matrix of each link relative to the previous link was calculated using the equation below:

$$ {}^{i}_{i}T = Rot_x(\alpha_i) * Trans_x(a_i) * Rot_z(\theta_i) * Trans_z(d_i) \quad (1) $$

where Rot_k is the rotational matrix about axis k, and $Trans_k$ is the translational matrix along axis k. With the transform matrices of each link relative to its previous link found, starting at the base and working towards the end link, each link transform matrix was then multiplied by each previous matrix in order to get the rotational matrix of each link in terms of the base.

Next the vector representing the center of mass for each link relative to itself was multiplied by that link's transform matrix relative to the base. This yielded a vector representing the location of the center of mass of reach link relative to the base. In order to obtain the velocity of each center of mass, the partial derivative of the position vector was taken in regards to $\partial\theta/\partial t$. An example set of calculations can be seen below.

$$ {}^{i}_{B}Pos_i = {}^{i}_{B}T_i * {}^{i}_{i}CoM \quad (2) $$

$$ {}^{i}_{B}Vel_i = \frac{\partial\theta}{\partial t}\left({}^{i}_{B}Pos_i\right) \quad (3) $$

Using the position and velocities of the center of mass, the kinetic and potential energy equations for each link could be solved for using the equation below.

$$ KE_k = \sum_{i=1}^{k} 0.5 * m_i * (v_{xi}^2 + v_{yi}^2 + v_{zi}^2) \quad (4) $$

$$ PE_k = \sum_{i=1}^{k} m_i * g * p_{yi} \quad (5) $$

where m_i is the mass of each link, v_i is the velocity of the centre of mass of each link in its respective direction, and p_{yi} is the position of the centre of mass of each link in the y direction. Once the kinetic and potential energies have been calculated, the Lagrange transform can be solved, as follows:

$$ \frac{d}{dt}\left(\frac{\partial KE}{\partial \dot{q}_i}\right) - \frac{\partial KE}{\partial q_i} + \frac{\partial PE}{\partial q_i} = U_i \quad (6) $$

where U_i is the torques applied to each link motor located at the origin, and q represents joint angles. For simplicity the equations of motion shown below are for a 2-link planar robot manipulator. The equations required for a three-dimensional, six-link robot manipulator begin to expand in length exponentially. The first equation of motion for a two-link manipulator in terms of applied torque are as follows:

$$ \begin{aligned} U_1 = {} & (L_1^2 * m_1 * \ddot{q}_1) + \\ & (2 * L_2^2 * m_2 * \ddot{q}_1 + L_2 * g * m_2 * \cos(q_1 + q_2)) + \\ & (L_1 * g * m_1 * \cos(q_1)) + (L_1 * g * m_2 + \cos(q_1)) - \\ & (2 * L_1 * L_2 * m_2 * \dot{q}_1^2 * \sin(q_2)) + \\ & (3 * L_1 * L_2 * m_2 * \ddot{q}_1 * \cos(q_2)) - \\ & (L_1 * L_2 * m_2 * \dot{q}_1 * \dot{q}_2 * \sin(q_2)) + 2 \end{aligned} \quad (7) $$

Next the simulation must generate the required trajectory information based on the user inputs. The simulation takes the 3-dimensional coordinates provided by the user and attempts to solve the required pose using a built-in inverse kinematics solver. Once all required poses have been solved, the simulation generates a time vector with evenly spaced intervals for each waypoint, with an additional 1 second interval before and after the last waypoint to allow smooth start-up and shut-down characteristics. The simulation then plots each set of required joint angles vs. time and uses a spline function to create a smooth path for each joint to follow. These values are used as desired joint positions as the simulation runs. In order to calculate desired velocities and accelerations, the derivative and double derivative of the position values are taken.

These values are then brought into the next subroutine of the simulation in order to simulate the dynamics of the robot and attempt to implement a control strategy. This subroutine first takes the equations of motion that were previously computed and substitutes the given characteristics of the robot (link masses, link lengths, centers of mass, etc.), then solves each equation in terms of joint acceleration. The simulation then calculates the error between each joint's desired position, velocity, and acceleration and the robot's simulated joint position, velocity, and acceleration. This information is then fed into a PID controller to attempt to generate the required torque needed to correct the error.

In order to implement a PID controller, an integrated and derived error term must be present. In order to achieve this, the PID is attempting to correct the velocity errors in each joint, with the integral and derivative terms being the joint position and acceleration, respectively. Once the PID calculates the required torque of each motor, that information is fed back into the system dynamics. The simulation takes the position and velocity of the previous time step, as well as the torque generated by the PID controller, and subs the values into the equations of motion for each link and solves in terms of joint acceleration. This acceleration term is then multiplied by the time constant of the simulation and added to the previous velocity term in order to obtain the current velocity. This process is then repeated to the current velocity term in order to calculate the current position. By linearizing each joint individually, a PID controller should be able to provide accurate control even though the overall system is non-linear.

During the simulation the gripper position will be constantly calculated in reference to a Cartesian coordinate system using forward kinematics. These values are then compared to the true trajectory calculated in the previous step. While this information is not necessary for the controller to operate, it provides value relatable information regarding the accuracy of the PID controller.

IV. Simulation Results

The PID controller that was implemented provided very promising results due to the fact that the system underwent individual linearization of each joint, and the PID treated each joint as an individual entity. Comparisons were made between both the individual joint performance, as well as the entire system overall. As the robot moved through the simulation, the desired trajectory and simulated trajectories were both plotted for comparison. Fig 1. below shows the traced path of the robot using the true trajectory, which is represented as the red line.

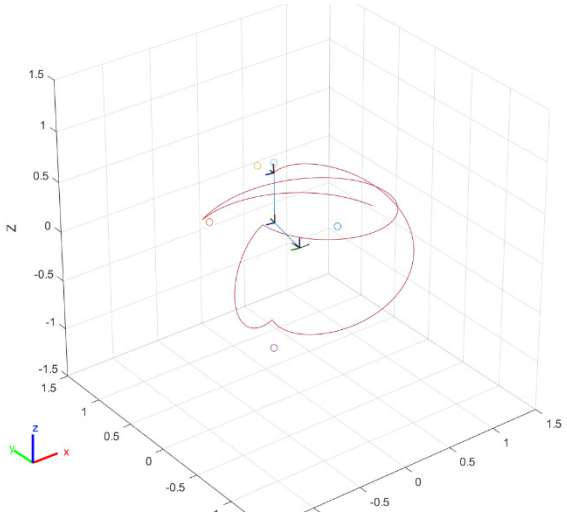

Figure 1. Desired Robot Trajectory

Fig 2. shows both the desired trajectory in red, as well as the simulated trajectory of the robot using the universally designed PID controller in blue.

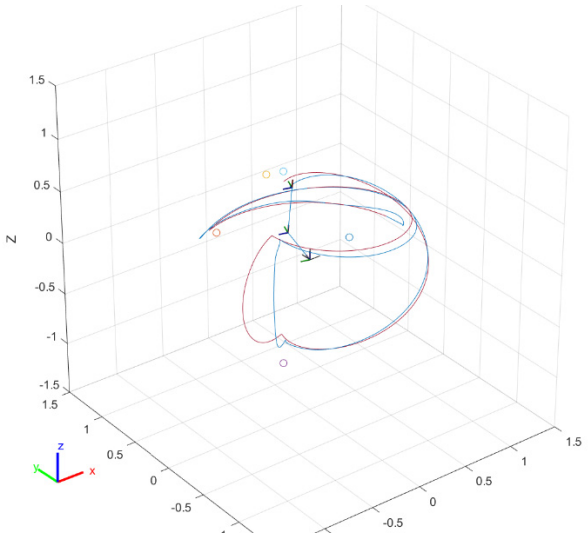

Figure 2. Desired and PID Controlled Robot Trajectory

In order to provide meaningful results, the linear distance between the end effector following the desired trajectory and simulated trajectory was constantly calculated. A comparison between the desired position and simulated position can also be found below in Fig 3.

Although the absolute error between the desired and controlled gripper position does not appear to be optimal, that is due mostly to the fact that a small error in each joint has a cascading effect along the manipulator arm with amplifies the errors by the time the end gripper position. This is evident when the individual PID controlled joint angles of each link are compared to the desired states. This can be seen in Fig 4.

It can be seen in Fig 4. that the control signal provided for each individual joint allowed the joint to track very closely to the desired trajectory. Through manual tuning of the PID, the proportional, integral, and derivative gains were found to be 150, 0.1. and 12, respectively. PID controllers are not typically designed to work with a non-linear system; however, this application using linearization showed strong results.

V. Conclusions

This brief paper outlined the need for accuracy in industrial robots, as well as a proposed approach to increasing accuracy in real-world systems. It outlined other approaches taken in the industry, as well as outlined a new approach to increasing repeatability accuracy. The proposed closed-loop feedback system was developed and tested on a virtual robotic system, and the results of individual joint tracking were shown, and a comparison of final end effector position was used to show overall effectiveness of the system. The results showed that even though there was slight error between the desired and controlled end effector position, the developed PID performed well to control each joint individually. Future work will look at expanding the results and applying it to an industrial robot that is used for experimentation.

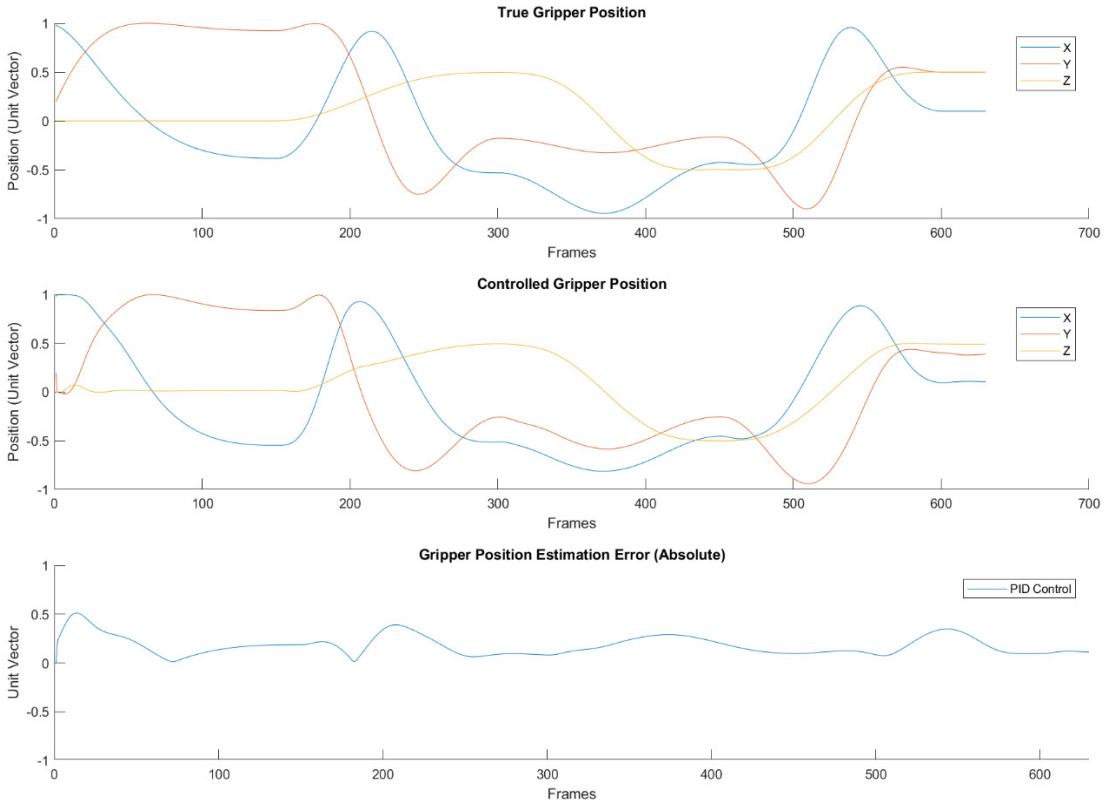

Figure 3. Desired and PID Controlled Robot Trajectory

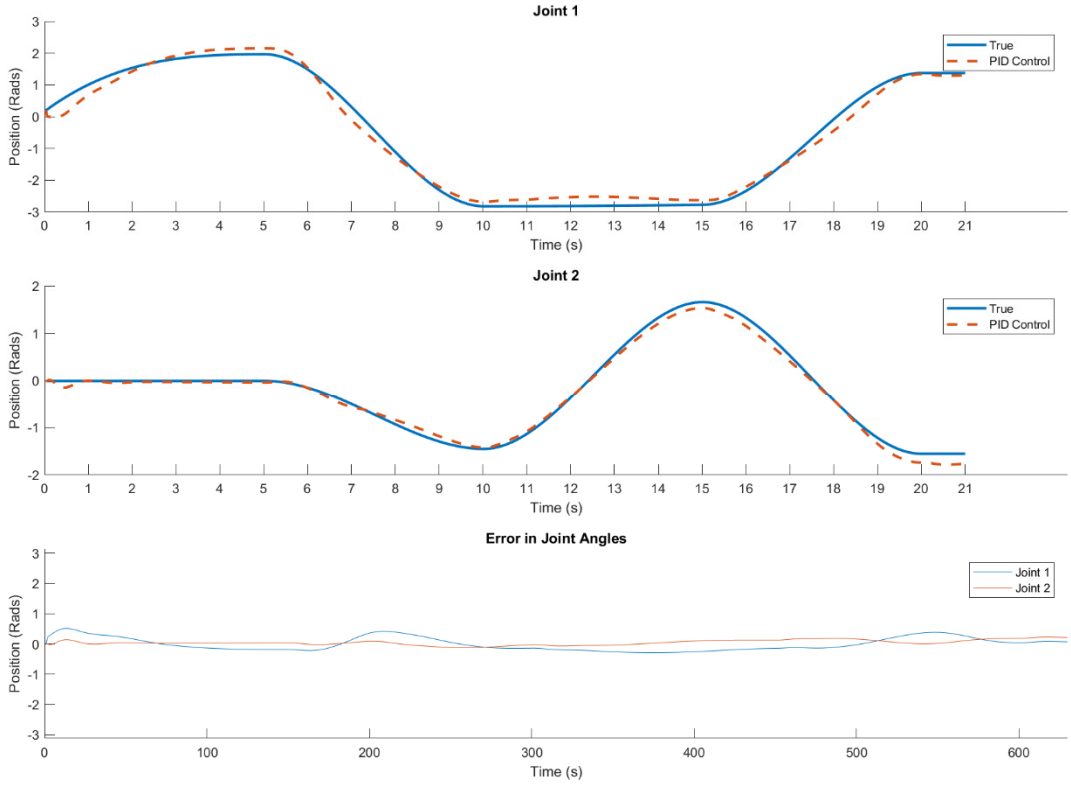

Figure 4. Individual Joint Comparisons

VI. REFERENCES

[1] J. Kim, K. Chang, B. Schwarz, A. S. Lee, M. Al-Shabi and S. A. Gadsden, "Dynamic Model and Motion Control of a Robotic Manipulator," *Journal of Robotics, Networking and Artificial Life,* vol. 4, no. 2, pp. 138-141, 2017.

[2] M. Al-Shabi, A. Cataford and S. A. Gadsden, "Quadrature Kalman Filters with Applications to Robotic Manipulators," in *IEEE International Symposium on Robotics and Intelligent Sensors*, Ottawa, Ontario, 2017.

[3] M. Cui, H. Liu, W. Liu and Y. Qin, "An Adaptive Unscented Kalman Filter-cased Controller for Simultaneous Obsticle Avoidance and Tracking of Wheeled Mobile Robots with Unknown Slipping Parameters," *Journal of Intelligent & Robotic Systems,* vol. 92, pp. 489-504, 2018.

[4] K. Joni, M. Ulum, T. Prasetyo and A. Y. Maulana, "Dynamic balancing umanoid robot using completmentary filter to optimized pid controller," *Journal of Physics: Conf.,* vol. 1211, 2019.

[5] A. V. Gulalkari, P. S. Pratama, G. Hoang, D. H. Kim, B. H. Jun and S. B. Kim, "Object tracking and following six-legged robot system using Kinect camera based on Kalman filter and backstepping controller," *Journal of Mechanical Science and Technology,* vol. 29, pp. 5425-5436, 2015.

[6] B. H. Prasetio, "Ensemble Kalman Filter and PID Controller Implementation on Self Balancing Robot," *International Electrioncs Symposium,* pp. 105-109, 2015.

[7] S. A. Gadsden, "Smooth Variable Structure Filtering: Theory and Applications," McMaster University (PhD Thesis), Hamilton, Ontario, 2011.

[8] S. A. Gadsden, S. R. Habibi and T. Kirubarajan, "Kalman and Smooth Variable Structure Filters for Robust Estimation," *IEEE Transactions on Aerospace and Electronic Systems,* vol. 50, no. 2, pp. 1038-1050, 2014.

[9] H. H. Afshari, S. A. Gadsden and S. R. Habibi, "Gaussian Filters for Parameter and State Estimation: A General Review and Recent Trends," *Signal Processing,* vol. 135, pp. 218-238, 2017.

[10] S. A. Gadsden, "An Adaptive PID Controller Based on Bayesian Theory," in *ASME Dynamic Systems and Control Conference*, Tysons Corner, Virginia, 2017.

[11] D. Morin, "Chapter 6," in *The Lagrangian Method*, 2006.

[12] Y. Yun, "Youngmok Yun: Roboticist in Austin," 21 02 2016. [Online]. [Accessed 10 07 2019].

[13] J. Onshaunjit and J. Srinonchat, "Nonlinear Kalman Filter Model for Balancing Optimization Gyro Robot," in *IEECON*, Krabi, Thailand, 2018.

[14] Martin, D. I. H. Putri, R. M. and C. Machbub, "Gait Controllers on Humanoid Robot Using Kalman Filter and PD Controller," in *Internation Conference on Control, Automation, Robotics and Vision*, Singapore, 2018.