Journal of Science and Technology
ISSN: 2362-1443 (Print) 3059-9490 (Online)
2024, Vol. 4, No. 2 (Special Issue), 82-89
DOI : https://doi.org/10.3126/jost.v4i2.78966

Control of Autonomous Robot Arm

Rahul Basu
Emeritus, Engineering
JNTU Bangalore, India
ORCID ID: Orcid:0000-0002-6179-1163

Article History:
Received: 30 July 2024
Revised: 12 August 2024
Accepted: 14 November 2024

Keywords— ***Autonomous, Feedback, Kalman Filter, robotic arm, Vision***

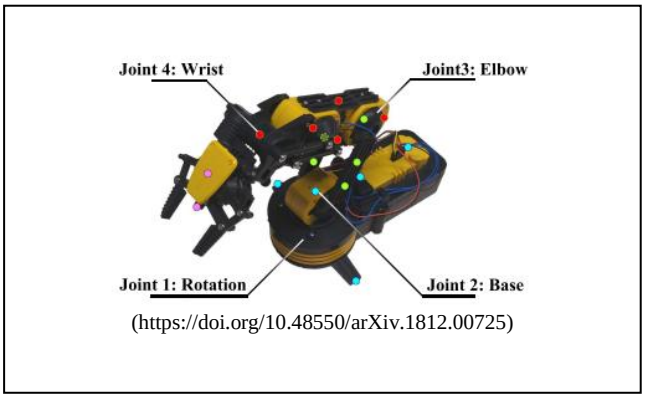
Abstract— **This paper explores the use of AI and ML technologies in space exploration, focusing on controlling robotic arms like the Puma robot arm using feedback technology. It discusses the application of Kalman Filters to a Robot arm augmented by real time feedback. The paper also highlights the OWI arm in the field of robotics and its potential impact on space exploration. The study underscores the potential of AI/ML technologies in space exploration and the role of low cost experiments in driving these advancements.**

I. INTRODUCTION

The advent of Artificial Intelligence (AI) and Machine Learning (ML) technologies has revolutionized various sectors, including space exploration. We focus on the application of these technologies in controlling robotic arms for space missions, specifically the Puma robot arm, using Bluetooth technology. Robotic arms have become an integral part of space missions, aiding in tasks ranging from sample collection to spacecraft repair. The integration of AI/ML technologies has further enhanced their capabilities, enabling autonomous functions and system-level capabilities. This paper discusses the application of programming by demonstration or imitation learning for trajectory planning of manipulators on free-floating spacecraft. The control of these robotic arms has been made more accessible and efficient through the use of Bluetooth technology. Recent advances by experimenters include the design and development of a Bluetooth Controlled Robot using Arduino, Bluetooth Module, and Motor Driver Module. It also explores the use of a dedicated Android app for controlling the robotic arm. The incorporation of AI using algorithms like cost of trajectory and weightage, vision sensors, Kalman filters and other aids is yet to be developed as projects for this low cost application. The ultimate goal is to lead to full autonomy for such devices, [1, 2]. The Taiwan OWI robotic arm is a low cost device that can be controlled using Bluetooth technology, which allows for wireless communication between the robot and a controlling device.

An Arduino Mega can control the OWI robot arm kit through a phone with Bluetooth arm.

This involves adding the user to the Bluetooth group for access and installing the necessary dependencies. [3].These methods highlight the flexibility and versatility of Bluetooth technology in controlling robotic arms like the Taiwan OWI. They also underscore the potential of such technology in advancing robotics and automation, particularly in applications like space exploration concepts. The unique aspects of the paper lie in the use of simple low cost tools to utilize advanced technological concepts.



(https://doi.org/10.48550/arXiv.1812.00725)

Fig. 1. The OWI Puma robot arm

The objective of this paper is to present a practical implementation of the Kalman filter algorithm for state estimation in dynamic systems. By illustrating the process through a step-by-step example using a simple constant acceleration model, the paper aims to demonstrate how the Kalman filter can effectively integrate noisy measurements to produce accurate state estimates. This will include a detailed explanation of the mathematical foundations, the algorithm's application, and the visualization of results through graphing routines. The ultimate goal is to provide a clear and comprehensive guide that serves as a reference for researchers and practitioners in fields such as robotics, navigation, and control systems.

II. BACKGROUND

In the realm of space exploration, the autonomy of robots is a critical area of focus. This study introduces a novel approach to trajectory planning for manipulators on spacecraft, utilizing programming by demonstration or imitation learning. A sophisticated 7-DoF robotic arm, attached to a compact spacecraft, is used for various tasks such as debris removal, on-orbit servicing, assembly, and autonomous docking [2]. The movement of the robotic arm influences the spacecraft's attitude, necessitating corrective actions from the Attitude Determination and Control System (ADCS). The proposed method identifies the optimal trajectory that minimizes these changes, thereby reducing the ADCS load. Power consumption, a crucial factor in spacecraft trajectory planning and control, is addressed by conducting trajectory learning offline. The process involves gathering data from demonstrations and encoding it into a probabilistic distribution of trajectories.

This distribution can then be used for planning in new situations, requiring minimal power for computations after deployment. The study also introduces a cost term to identify the optimal trajectory that minimizes attitudinal changes. This comprehensive approach to trajectory planning could pave the way for more efficient and autonomous space missions in the future.

The paper gives special attention to the Taiwan OWI, a significant player in the field of robotics. Despite the political complexities surrounding Taiwan, it continues to make strides in technological advancements. The paper aims to shed light on how these advancements can contribute to the broader field of space exploration and other fields.

A. Trajectory Calculation:

Calculation involves moving a robotic arm from an initial to a set position by changing the joint angle. This is crucial in modern applications like bin-picking, where the goal is to pick randomly placed objects. The process uses modified trajectory metrics to compute collision-free trajectories. The use of the PSO (Particle swarm optimization) is described for trajectory planning by [4]. Vision based trajectory planning is described by [5].

B. Predictor-Corrector Algorithm:

Kalman Filter Tools are powerful algorithms used for estimating and predicting system states in uncertain conditions, widely used in applications like target tracking, navigation, and control. In robotic arms, they estimate the system state (e.g., arm position and velocity) based on observed measurements over time. The filter combines the predicted system state and the latest measurement in a weighted average, producing more precise estimates. They are widely used in the field of robotics, particularly for autonomous robots and robot arms, for state estimation [6, 7]. State estimation is the problem of accurately determining variables about your robot, such as its position with respect to some global frame, velocity, acceleration, IMU biases, and other dynamical variables, given the robot's sensors and physics kinematics. The standard Kalman Filter works well for linear systems [7]. However, many systems in robotics are nonlinear [6]. To handle these nonlinear systems, the Extended Kalman Filter (EKF) is often used. The EKF is a nonlinear full-state estimator that approximates the state estimate with the lowest covariance error when given the sensor measurements, the model prediction, and their variances [6].

For instance, in autonomous mobile robot competitions, accurate localization is crucial for creating an autonomous competition robot. Two common localization methods are odometry and computer vision landmark detection. Odometry provides frequent velocity measurements, while landmark detection provides infrequent position measurements. The state can also be predicted with a physics model. These three types of localization can be "fused" to create a more accurate state estimate using an Extended Kalman Filter (EKF).

In recent years, there have been significant advancements in the field of robotics, such as Boston Dynamics' Spot, Da Vinci surgical robot, Tesla's Autopilot, and more. These advancements have further highlighted the importance of state estimation and the role of Kalman Filters. Kalman Filters and EKFs do have limitations and assumptions that need to be considered when implementing them in real-world applications. For example, error in velocity will accumulate in position when using odometry. Therefore, it's crucial to understand these limitations and assumptions to effectively use these tools in robotics. The Kalman Filter is based totally on a set of mathematical formulas that iteratively update estimates primarily based on new measurements. The following is a simplified rationalization of the fundamental equations within the Kalman Filter:

The Kalman Filter is a recursive algorithm used for estimating the state of a dynamic system from a series of noisy measurements. It is widely used in control systems, robotics, navigation, and more. The algorithm consists of two main steps: the Prediction Step and the Update Step.

1) Prediction Step

a) State Prediction:

$$x\hat{}k \mid k-1 = Fx\hat{}k-1 \mid k-1 + Buk-1 \quad (1)$$

$\hat{x}_{k|k-1}$: The predicted state at time k. This is our best guess of the system's state before incorporating the new measurement at time k.

F: The state transition matrix, which models how the system evolves from one time step to the next.

$\hat{x}_{k-1|k-1}$: The estimated state at the previous time step (k-1), after incorporating the measurement from that time step.

B: The control input matrix, which relates the control input to the change in state.

u_{k-1} : The control input at time (k-1), representing external influences or actions taken on the system.

b) Covariance Prediction:

$$Pk \mid k-1 = FPk-1 \mid k-1FT + Q \quad (2)$$

$P_{k|k-1}$: The predicted state covariance at time k, which represents the uncertainty in the predicted state.

$P_{k-1|k-1}$: The estimated state covariance at the previous time step k-1.

Q: The process noise covariance, representing the uncertainty introduced by the model (e.g., due to approximations or unknown influences).

The State Prediction equation predicts the current state based on the previous state and the control input.

The Covariance Prediction equation predicts the uncertainty of the current state by considering the uncertainty from the previous state and the process noise.

2) Update Step

a) Kalma Gain:

$$Kk = Pk \mid k-1HT(HPk \mid k-1HT + R) - 1 \quad (3)$$

K_k: The Kalman Gain, which determines how much weight to give to the new measurement versus the prediction.

H: The observation matrix, which relates the state to the measurement.

R: The measurement noise covariance, representing the uncertainty in the measurements.

$P_{k|k-1}$: The predicted state covariance from the prediction step.

3) State Update:

$$x\hat{}k \mid k = x\hat{}k \mid k-1 + Kk(zk - Hx\hat{}k \mid k-1) \quad (4)$$

$\hat{x}_{k|k}$: The updated state estimate at time k, which combines the prediction with the new measurement.

z_k: The measurement at time k.

$H\hat{x}_{k|k-1}$: The predicted measurement, based on the predicted state.

4) Covariance Update:

$$Pk \mid k = (I - KkH)Pk \mid k-1 \quad (5)$$

Pk|k: The updated state covariance at time \(k\), representing the updated uncertainty after incorporating the new measurement.

I: The identity matrix, ensuring the correct dimensionality.

The Kalman Gain determines the optimal blending of the predicted state and the measurement.

The State Update refines the prediction using the new measurement, yielding a more accurate estimate.

The Covariance Update adjusts the uncertainty of the state estimate, reflecting the incorporation of the new information.

Together, the Prediction and Update steps of the Kalman Filter recursively refine the estimate of the system's state, balancing the predicted dynamics with incoming measurements. Kalman filter tools for robot arms using low cost 3 D Xbox 360 cameras are described by Berti et. al and Welch [8, 9].

In Fuzzy-Based Processing, a methodology grounded in logic and utilizing fuzzy rules is employed to rectify positioning inaccuracies. Typically, a human operator leverages their binocular vision to adjust the tool and rectify positional errors. This is commonly observed in scenarios such as medical procedures or nuclear tool handling [10]. However, in situations like bomb disposal or archaeological explorations, the robot must be equipped with binocular vision. Additionally, an AI-driven set of rules is crucial for successful operation. [11]. These rules, guided by artificial intelligence, enable the robot to make decisions and learn from experience, thereby enhancing the precision and success rate of its tasks. [12, 13, 14, 15]. There are several alternatives to Kalman Filters, each with its own strengths and limitations depending on the specific application. The Kalman Filter, Particle Filter, and Extended Kalman Filter are state estimation techniques used in various applications. The Kalman Filter is suitable for linear models with Gaussian state distributions and is computationally efficient, making it popular in finance, control, and navigation. The Particle Filter handles nonlinear models and arbitrary state distributions but is computationally expensive, especially for high-dimensional systems, and is particularly useful in robotics and tracking. The Extended Kalman Filter, designed for mildly nonlinear models, assumes Gaussian distributions and offers improved accuracy over the standard Kalman Filter for such systems. It is commonly used in finance, sensor fusion, and robotics. The choice of filter depends on the system's nature (linear or nonlinear), noise type (Gaussian or non-Gaussian), computational resources, and required accuracy.

The following types of filters can be used:

Extended Kalman Filter (EKF): This is used for mildly nonlinear systems. The EKF is a nonlinear version of the Kalman Filter that linearizes about an estimate of the current mean and covariance, [13].

Particle Filter: This is used for highly nonlinear and non-Gaussian systems. Particle filters represent the posterior distribution of a state by a set of random samples, or particles, and their weights.

Unscented Kalman Filter (UKF): The UKF addresses the limitations of the EKF by using a deterministic sampling technique known as the unscented transformation to pick a minimal set of sample points (called sigma points) around the mean.

Ensemble Kalman Filter (EnKF): The EnKF uses a Monte Carlo approach to deal with the nonlinearity of systems. Alpha-Beta Filter: This is a simpler form of the Kalman Filter. Double Exponential Smoothing: This method is used for predictive tracking of user position and orientation. It runs approximately 135 times faster with equivalent prediction performance and simpler implementations compared to Kalman and extended Kalman filter-based predictors. [14]. Sigma Pointer Filters that avoid the use of Jacobian matrices was described by ElShabi [16]. Vision sensors are a crucial addition to robotic arms like the OWI-535, enabling them to interact more effectively with their environment. These sensors can provide complementary information in sensor-equipped robotic systems, [17]. Binocular enhancement in estimating position is described in [18]. A review of the use is described in [19].

III. RESULTS AND DISCUSSION

A. *Vision Sensing*

The addition of vision sensors to the OWI robot arm can enhance its capabilities in various ways. For instance, it can help in real-time status prediction using an external camera, thus allowing researchers to control them with high flexibility. In another application, with the Open MV camera, the robot arm could pick up a red cube and place it in a fixed position [20]

B. *Obstacle Avoidance*

Obstacle avoidance is a critical aspect of robotic systems, including the Taiwan OWI arm. Some algorithms that have been used for obstacle avoidance in robotic arms: *Improved RRT Algorithm*: One method involves an improved version of the Rapidly-exploring Random Tree (RRT) algorithm..[21. *Improved Artificial Potential Field Algorithm:* Another approach is based on *the Artificial Potential Field* (APF) algorithm., and sensor network based algorithm.[22,23]. These algorithms can be adapted and applied to the Taiwan OWI arm for effective obstacle avoidance.

C. *Methodology.*

Obstacle Avoidance can be integrated into the vision system which can normally usually only determine 2D offsets instead of 3D positions [24]. However, it works for targets at rest or targets moving on the conveyor belt with a uniform speed. In the CRAVES system [17], a 3D model is used to create a large number of synthetic data, train a vision model in this virtual domain, and apply it to real-world images after domain adaptation.

D. *Application To A Simple Example*

Given below is the output of a simple program describing an acceleration model. It depicts the behaviors of the Kalman Filter written in PYTHON. The program initializes a Kalman filter and applies it to a series of position measurements. The state estimate is updated after each measurement, showing the filter's ability to predict and correct the state based on the process and measurement models. In the results (fig2,3): The green line represents the true positions, red crosses represent the noisy measurements. The blue line represents the Kalman filter's estimates.

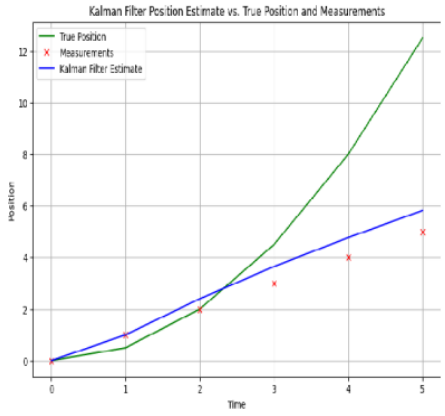


Fig. 2. Kalman Filter applied to simple acceleration model

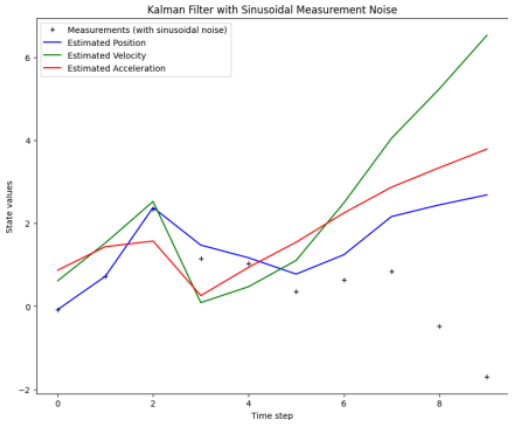


Fig. 3. Kalman Filter with sinusoidal noise

1) **Graphing Routine**: This routine uses "**matplotlib"** to plot the true positions, measurements, and Kalman filter estimates.

2) **State Estimates**: It stores the state estimates after each measurement.

3) **True Positions**: It generates true positions for comparison, assuming a simple constant acceleration model for simplicity.

4) **Measurement Positions**: It stores the measurement positions

The graph will help to visualize how the Kalman filter estimates the true state over time, despite the noisy measurements. (Numerical values are listed in the Appendix). The use of the Kalman Filter in the context of simple robot arms is sparsely reported in the literature. A few reports has appeared comparing the true position with the error in [25-29]. There is no doubt that Space Agencies like NASA etc have done extensive work on these technologies, however the reports are difficult to find. Some experimental simulations have been given comparing the error to the true path (see fig.4).

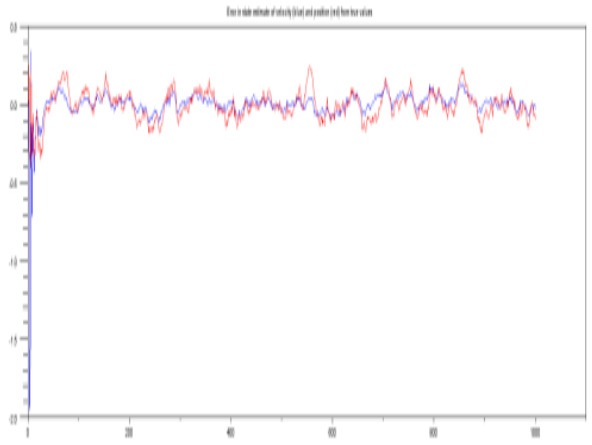


Fig. 4. Error vs True Position [25]

IV. CONCLUSION

There is potential of AI/ML technologies in transforming space exploration, particularly through the use of robotic arms and Bluetooth control systems. It also highlights the role of regions like Taiwan in driving these technological advancements. The findings could pave the way for more sophisticated and autonomous space missions in future. Such applications utilize indigenous tools and low cost technology readily available/capable of development, and thus contribute to self reliance in the field of space technology.

This paper gives a detailed implementation and analysis of the Kalman filter, a powerful tool for state estimation in dynamic systems. Through a simple example involving a constant acceleration model, it has been demonstrated how the Kalman filter integrates noisy measurements to produce accurate and reliable state estimates. The paper covered the mathematical foundations of the Kalman filter, step-by-step implementation, and visualization of the results, emphasizing its effectiveness and practical applications.

The implementation showed that the Kalman filter could improve state estimates even when measurements are noisy or uncertain. The graphing routine further illustrated how the filter adapts to new measurements, continuously refining the state estimates over time. This makes the Kalman filter an invaluable asset in fields such as robotics, navigation, and control systems where accurate state estimation is crucial.

In conclusion, the Kalman filter remains a cornerstone of modern estimation theory, offering robust performance in diverse applications. This paper serves as a comprehensive guide for researchers and practitioners, highlighting the filter's practical utility and providing a foundation for further exploration and application in advanced dynamic systems.

REFERENCES

[1] I. A. Nesnas, L. M. Fesq, and R. A. Volpe, "Autonomy for Space Robots: Past, Present, and Future," Curr Robot Rep, vol. 2, pp. 251–263, 2021.

[2] A. Shyam RB, Z. Hao, U. Montanaro, S. Dixit, A. G. Rathinam, Y. Gao, G. Neumann, and S. Fallah, "Autonomous Robots for Space: Trajectory Learning and Adaptation Using Imitation," Front. Robot. AI, vol. 8, p. 638849, 2021.

[3] [Online]. Available: https://orionrobots.co.uk/2021/01/23/bluedot-usb-arm-control.html

[4] X. Miao, H. Fu, and X. Song, "Research on motion trajectory planning of the robotic arm of a robot," Artif Life Robotics, vol. 27, pp. 561–567, 2022.

[5] S. Chen, J. Zhang, and T. Zhang, "Fuzzy Image Processing Based on Deep Learning: A Survey," in The International Conference on Image, Vision and Intelligent Systems (ICIVIS 2021), Lecture Notes in Electrical Engineering, vol. 813, J. Yao, Y. Xiao, P. You, and G. Sun, Eds. Springer, Singapore, 2022. [Online]. Available: https://doi.org/10.1007/978-981-16-6963-7_10

[6] E. Kou and A. Haggenmiller, "Extended Kalman Filter State Estimation for Autonomous Competition Robots," [Online]. Available: https://github.com/BubblyBingBong/EKF

[7] H. Wang, "Fuzzy control system for visual navigation of autonomous mobile robot based on Kalman filter," Int J Syst Assur Eng Manag, vol. 14, pp. 786–795, 2023. [Online]. Available: https://doi.org/10.1007/s13198-021-01570-5

[8] E. Berti, A.-J. Sánchez-Salmerón, and F. Benimeli, "Kalman Filter for Tracking Robotic Arms Using low cost 3D Vision Systems," in ACHI 2012 - 5th International Conference on Advances in Computer-Human Interactions, 2012.

[9] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," in SIGGRAPH, 2001.

[10] O. Gomes, "I, Robot: the three laws of robotics and the ethics of the peopleless economy," AI and Ethics, vol. 4, 2023. [Online]. Available: https://doi.org/10.1007/s43681-023-00263-y

[11] R. Czabanski, M. Jezewski, and J. Leski, "Introduction to Fuzzy Systems," in Theory and Applications of Ordered Fuzzy Numbers, P. Prokopowicz, J. Czerniak, D. Mikołajewski, Ł. Apiecionek, and D. Ślęzak, Eds. Springer, Cham, vol. 356, 2017. [Online]. Available: https://doi.org/10.1007/978-3-319-59614-3_2

[12] X. Yu, Z. Fan, H. Wan, Y. He, J. Du, N. Li, Z. Yuan, and G. Xiao, "Positioning, Navigation, and Book Accessing/Returning in an Autonomous Library Robot using Integrated Binocular Vision and QR Code Identification Systems," Sensors, vol. 19, p. 783, 2019. [Online]. Available: https://doi.org/10.3390/s19040783

[13] J. J. LaViola Jr., "Double Exponential Smoothing: An Alternative to Kalman Filter-Based Predictive Tracking," in International Immersive Projection Technologies Workshop Eurographics Workshop on Virtual Environments, A. Deisinger and A. Kunz, Eds., 2003.

[14] N. Kumari, R. Kulkarni, M. R. Ahmed, and N. Kumar, "Use of Kalman Filter and Its Variants in State Estimation: A Review," in Artificial Intelligence for a Sustainable Industry 4.0, S. Awasthi, C. M. Travieso-González, G. Sanyal, and D. Kumar Singh, Eds. Springer, Cham, 2021. [Online]. Available: https://doi.org/10.1007/978-3-030-77070-9_13

[15] C. Suliman, C. Cruceru, and F. Moldoveanu, "Mobile Robot Position Estimation Using the Kalman Filter," Scientific Bulletin of the Petru Maior University of Tirgu Mures, vol. 6 (XXIII), pp. 75-78, 2009.

[16] M. Al-Shabi, "Sigmaa Point Filters in Robotic Applications," Intelligent Control and Automation, vol. 6, no. 3, pp. 168-183, 2015. [Online]. Available: https://doi.org/10.4236/ica.2015.63017

[17] Y. Zuo, W. Qiu, L. Xie, et al., "CRAVES: Controlling Robotic Arm with a Vision-based Economic System," 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1812.00725

[18] W. P. Ma, W. X. Li, and P. X. Cao, "Binocular Vision Object Positioning Method for Robots Based on Coarse-fine Stereo Matching," Int. J. Autom. Comput., vol. 17, pp. 562–571, 2020. [Online]. Available: https://doi.org/10.1007/s11633-020-1226-3

[19] N. Kumari, R. Kulkarni, A. Riyaz, and N. Kumar, "Use of Kalman Filter and Its Variants in State Estimation: A Review," in Artificial Intelligence for a Sustainable Industry 4.0, S. Awasthi, C. M. Travieso-González, G. Sanyal, and D. Kumar Singh, Eds. Springer, Cham, 2021.

[20] [Online].Available: https://content.instructables.com/pdfs/EIZ/17ED/JCAUIXSV/An-Affordable-Vision-Solution-With-Robot-Arm.pdf

[21] H. Zhang, Y. Zhu, X. Liu, and X. Xu, "Analysis of Obstacle Avoidance Strategy for Dual-Arm Robot Based on Speed Field with Improved Artificial Potential Field Algorithm," Electronics, vol. 10, p. 1850, 2021. [Online]. Available: https://doi.org/10.3390/electronics10151850

[22] H. Zhang, Y. Zhu, and X. Liu, "Analysis of Obstacle Avoidance Strategy for Dual-Arm Robot Based on Speed Field with Improved Artificial Potential Field Algorithm," Electronics, vol. 10, p. 1850, 2021. [Online]. Available: https://doi.org/10.3390/electronics10151850

[23] W. Shi, K. Wang, and C. Zhao, "Control of Robotic Arm Using Visual Feedback," Applied Sciences, vol. 12, no. 8, p. 4087, 2022. [Online]. Available: https://doi.org/10.3390/app12084087

[24] L. Chen, H. Yang, and P. Liu, "Intelligent Robot Arm: Vision-Based Dynamic Measurement System for Industrial Applications," in Intelligent Robotics and Applications. ICIRA 2019. Lecture Notes in Computer Science, vol. 11744, H. Yu, J. Liu, L. Liu, Z. Ju, Y. Liu, and D. Zhou, Eds. Springer, Cham. [Online]. Available: https://doi.org/10.1007/978-3-030-27541-9_11

[25] 25.[Online].Available: https://dsp.stackexchange.com/questions/8860/kalman-filter-for-position-and-velocity-introducing-speed-estimates

[26] Y. Zhao and Y. Jia, "Extended Kalman Filter Calibration Method Based on Lightweight Robotic Arm," in Proceedings of 2021 Chinese Intelligent Systems Conference, Y. Jia, W. Zhang, Y. Fu, Z. Yu, and S. Zheng, Eds., vol. 805, Lecture Notes in Electrical Engineering. Singapore: Springer, 2022. pp. 685-696. doi: https://doi.org/10.1007/978-981-16-6320-8_64

[27] S. Horvath and H. Neuner, "System identification of a robot arm with extended Kalman filter and artificial neural networks," Journal of Applied Geodesy, vol. 13, no. 2, pp. 135-150, 2019. doi: https://doi.org/10.1515/jag-2018-0045

[28] G. Rodriguez, "Kalman Filtering, Smoothing and Recursive Robot Arm Forward and Inverse Dynamics," NASA Technical Report, 1987. [Online]. Available: https://ntrs.nasa.gov/api/citations/19870008018/downloads/19870008018.pdf

[29] M. H. Lashari et al, "Kalman Filtering Techniques and Applications," arXiv, Jun. 2024. [Online]. Available: https://arxiv.org/abs/2406.04503

APPENDICES

(values generated by running simulations)

Appendix A:

Program for simulating the Kalman Filter with a simple acceleration model

```
import numpy as np

class KalmanFilter:
    def __init__(self, A, B, H, Q, R, P, x):
        self.A = A  # State transition matrix
        self.B = B  # Control input matrix
        self.H = H  # Observation matrix
        self.Q = Q  # Process noise covariance
        self.R = R  # Measurement noise covariance
        self.P = P  # Estimate error covariance
        self.x = x  # State estimate

    def predict(self, u):
        self.x = np.dot(self.A, self.x) + np.dot(self.B, u)
        self.P = np.dot(np.dot(self.A, self.P), self.A.T) + self.Q

    def update(self, z):
        y = z - np.dot(self.H, self.x)
        S = np.dot(self.H, np.dot(self.P, self.H.T)) + self.R
        K = np.dot(np.dot(self.P, self.H.T), np.linalg.inv(S))
        self.x = self.x + np.dot(K, y)
        I = np.eye(self.A.shape[0])
        self.P = np.dot(np.dot(I - np.dot(K, self.H), self.P), (I - np.dot(K, self.H)).T) + np.dot(np.dot(K, self.R), K.T)

# Sample input data
A = np.array([[1, 1], [0, 1]])  # State transition matrix
B = np.array([[0.5], [1]])      # Control input matrix
H = np.array([[1, 0]])          # Observation matrix
Q = np.array([[1, 0], [0, 1]])  # Process noise covariance
R = np.array([[1]])             # Measurement noise covariance
P = np.array([[1, 0], [0, 1]])  # Estimate error covariance
x = np.array([[0], [0]])        # Initial state estimate

kf = KalmanFilter(A, B, H, Q, R, P, x)

# Control input (acceleration)
u = np.array([[2]])

# Measurements (positions)
measurements = [1, 2, 3, 4, 5]

print("Initial state:")
print(kf.x)

# Apply Kalman filter
for i, z in enumerate(measurements):
    kf.predict(u)
    kf.update(np.array([[z]]))
    print(f"State estimate after measurement {i + 1}:")
    print(kf.x)
```

(Note: graphing routine is excluded).

Numerical Output plotted in Graph (Fig 1)

```
Initial state:
[[0]
 [0]]
State estimate after measurement 1:
[[0.66666667]
 [1.33333333]]
State estimate after measurement 2:
[[1.38461538]
 [2.30769231]]
State estimate after measurement 3:
[[2.1875]
 [3.125]]
State estimate after measurement 4:
[[3.05769231]
 [3.84615385]]
State estimate after measurement 5:
[[3.98947368]
 [4.47368421]]
```

Appendix B:

The following is a practical implementation of the Kalman filter algorithm for state estimation in a simple constant acceleration model. This example illustrates how the Kalman filter can be used to integrate noisy measurements to produce accurate state estimates.

1. Model Setup

Consider a system where an object is moving with constant acceleration. The state vector includes position, velocity, and acceleration:

$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix}$$

The system dynamics can be described by the following state-space model:

$$\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \mathbf{B}u_k + \mathbf{w}_k, \quad \mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k$$

Where:

X_k is the state vector at time step (k) F is the state transition matrix- B is the control input matrix, - uk is the control input (acceleration), wk is the process noise, - zk is the measurement vector, - H is the measurement matrix, - vk is the measurement noise.

2. State Transition and Measurement Matrices

For the constant acceleration model, the state transition matrix F and the measurement matrix H can be defined as follows:

$$\mathbf{F} = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}$$, H= [1 0 0]

Where Delta t is the time step. The control input matrix B is:

$$\mathbf{B} = \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \\ 1 \end{bmatrix}$$

3. Kalman Filter Equations

Prediction Step

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}u_{k-1},$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}\mathbf{P}_{k-1|k-1}\mathbf{F}^T + \mathbf{Q}$$

Update Step

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}^T(\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T + \mathbf{R})^{-1}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1})$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\mathbf{P}_{k|k-1}$$

Where: $\hat{\mathbf{x}}_{k|k-1}$ is the predicted state, $\mathbf{P}_{k|k-1}$ is the predicted covariance, - Kk is the Kalman Gain

$\hat{\mathbf{x}}_{k|k}$ is the updated state estimate

$\mathbf{P}_{k|k}$ is the updated covariance

4. Example Implementation:

Assume an initial state of rest (position = 0, velocity = 0, acceleration = 0), with a time step Delta t of 1 second. Let's simulate 10 time steps with a constant control input (acceleration) of 1 m/s², and process and measurement noise with zero mean and small covariance.