Improving Indoor Tracking Accuracy through Sensor Fusion: A Low-Cost, Neural Network-Assisted Visual System for Real-time Position Estimation

Houlin Chen[1], Lu Liang[2] and Lei Wang[3]

University of Wisconsin-La Crosse, Wisconsin WI 54601, USA
houlinchen@outlook.com
lianglu99es@gmail.com
lwang@uwlax.edu

Abstract. GPS is commonly used in outdoor situations due to its strong satellite system. However, GPS encounters difficulties in indoor contexts, where the effects of signal blocking and fading can lead to false readings. Using neural networks, this study presented a cost-effective and autonomous model of a visual tracking system. Using the ArUco code as a starting point, the system was optimized by employing the YOLO neural network to increase the accuracy of determining position. The integration of an inertial measurement unit (IMU) with a vision-based position estimate was done to achieve seamless tracking results. A Kalman filter model is introduced to predict and update detection values in sensor fusion to further optimize the system. This study is based on the principles of visual localization theory and coordinate system dimensional transformation and integrates a sensor substitution estimation system. The study refines the dataset and incorporates a large experimental set to validate the performance of the new system. The experiments comprehensively evaluate the capability and practical prospects of the new system in terms of accuracy, low cost, interference immunity, reliability, and real-time. The new system provides enhanced awareness and higher processing efficiency for different types of detection data with the help of neural networks. The results show that this optimized combination significantly improves the detection capability of the system under complex conditions. It is remarkable that this approach has a minimal additional cost beyond the sensors and intelligent platform and does not rely on any costly technologies such as optoelectronic or radiolocation.

Keywords: Sensor fusion, Monocular vision, Neural network, Indoor localization, IMU, Kalman filter

1. Introduction

Blocked by dense and complex building materials in cities coupled with the fading effect of the signal, GPS performance for the accurate positioning of indoor objects is unsatisfactory. This long-standing structural problem in indoor positioning has led to the development of indoor positioning techniques based on different principles. From thermal imaging reflectors and military radars to bionic acoustic detectors, there is a proliferation of technologies with remote height-tracking capabilities. With the rise and development of smart terminals in the new century, there has been a gradual shift towards civilian, low-cost, and portable indoor positioning tools. Indoor positioning systems based on smart terminals or mobile devices are widely used in new intelligent living scenarios, such as drone navigation, residential security, and virtual reality, among others [1]-[3]. In the spirit of affordability and portability, wireless LAN and Bluetooth are widely used for indoor positioning due to their signal wave principle. The former system, known as WPS, requires active localization via mobile devices connected to a hotspot. The latter involves the interconnection of devices. They are an important class of smart terminals that require a known signal source in advance and ensure that the device is within the source, such as the Bluetooth tag and the location of the WLAN [4] [5]. Another example is radio frequency identification (RFID), which requires existing RF databases. Or with depth cameras and radar, but this inevitably drives up the cost [6]. Integrating multiple technologies to improve system-wide stability and accuracy is a proven approach. A good example of this is the use of Google Maps. To compensate for the shortcomings of GPS, access to the user's Bluetooth and wireless LAN and the uploading of a rough floor plan of the building by the user are used for multi-point comparison and triangulation to obtain a rough indoor location [7]. In practice, however, this presents several challenges such as lack of accuracy and privacy of civil buildings [8].

Monocular vision-based, or vision-based, indoor positioning techniques refer to the extraction of information from two-dimensional images to measure the location in a three-dimensional coordinate system [9]. This localization method is straightforward and relatively accurate for indoor tracking. It does not rely on external signal sources or knowledge of the building to estimate the location of the observation point. And any camera-equipped smart terminal is capable of achieving this goal. This meets the basic requirements of low cost and portability. However, it has been criticized for its drawbacks. One of these is the over-reliance on the capture capability of the camera itself. This means that tracking will be completely interrupted when environmental conditions are poor, such as low light or obstruction [10]. In addition, the large number of images that must be tracked continuously in the form of video frames places a significant burden on image processing. This can lead to problems with real-time performance not being guaranteed [11]. This creates resistance to the diffusion of this convenient and potential technology.

In order to address the limitations associated with conventional visual localization algorithms, such as their susceptibility to interference and limited real-time capabilities, our approach introduces a novel visual localization method based on deep learning. By combining neural networks and sensor fusion, we have devised an indoor localization system that offers improved anti-interference capability and enhanced real-time performance. Leveraging object detection powered by neural networks, we have significantly enhanced the accuracy of visual localization while simultaneously achieving efficient real-time feedback. The system is specifically designed to operate on smart mobile platforms, paving the way for its widespread implementation in diverse indoor smart environments in the coming years.

To translate two-dimensional (2D) image projections into three-dimensional (3D) points in real-world surroundings, we leverage the employment of ArUco codes to assist with the localization task. A remarkable advantage of this particular marker lies in its capability to furnish sufficient correspondence information to compute the camera's pose [12]. Furthermore, owing to the ArUco code's internal binary encoding, the marker maintains distinct stability regarding checks and corrections. To enhance the accuracy of marker detection in challenging scenarios involving rapid motion and fluctuations in lighting conditions, we employ neural networks. Occasionally, situations may arise during marker detection where the marker remains elusive due to the camera's limited field of view. In such cases, we utilize the high acquisition frequency of the cell phone sensor to bridge this gap. Additionally, we introduce Kalman filtering to rectify the accumulation of errors in the sensor data. The localization process commences with the detection of markers appearing in each video frame, providing us with crucial information such as the marker's ID and the 2D coordinates of each corner. The 3D coordinates of the camera are deduced by solving the Perspective-n-Point (PnP) problem. The sensors in the system continue to function. The proposed system integrates the accelerometer and gyroscope which allows for the determination of the current position. Consequently, this outcome serves to fill in the gaps where no marker detection occurs. Ultimately, based on the calculated 3D coordinates of the device, we can plot the device's trajectory and make comparisons with other systems.

The main body of this paper is divided into two parts including system design and system testing experiments. The system design is divided into seven subparts in logical order: general overview of system structure, camera calibration calculation, YOLO detection, camera pose estimation, Motion Tracking Correction, EPnP algorithm, and data set processing. The system test experiments are divided into four subparts in logical order: experimental design and requirements, test sets for the original systems, test sets for the optimized systems, result analysis, and comparison. The evaluation of the proposed system and conclusion will be followed at the end.

2. System Design

2.1. Overview

The aim of the proposed system is to track and localize objects that are in 3D space and to perform tracing point plotting in real-time. The parameters of the proposed system come from two main sources: the first one is the camera visual capture frame of the mobile device, which should be considered as high frequency uninterrupted. The second aspect comes from the detection readings of the sensors.

The new system is divided into five units according to different divisions of labor. Figure 1 shows the structure design and modules of the system. Each unit in turn includes specific sub-module elements. The first module of the system is camera calibration, where parameters such as camera pose, and distance are calculated by transforming the coordinate system to assess the initial position of the mobile device in 3D space. YOLO acts as a neural network-based single-stage detector that accepts the input from module one and starts detecting the target. This is module two. Its output is the target ID and planar coordinate parameters of the subject. Module three appears as an alternative auxiliary module. It is based on the IMU to estimate the distance and trajectory of the device from the sensor readings when vision-based detection is not available. The second step of Module III is to input the sensor readings into the Kalman filter. The Kalman filter corrects for errors in the previous step with linear predictions and updates. This module will continue until the vision-based detection has re-detected the target. At this point, the system moves successively to modules four and five. Module four focuses on the conversion of the detection results using the EPnP algorithm to calculate the distance and position parameters of the camera. Module 5 consists of a 3D coordinate system and a point plotting system to plot the trajectory in real-time. These trajectories are then displayed on the system screen in real-time. These modules will be analyzed in the following sections one by one.

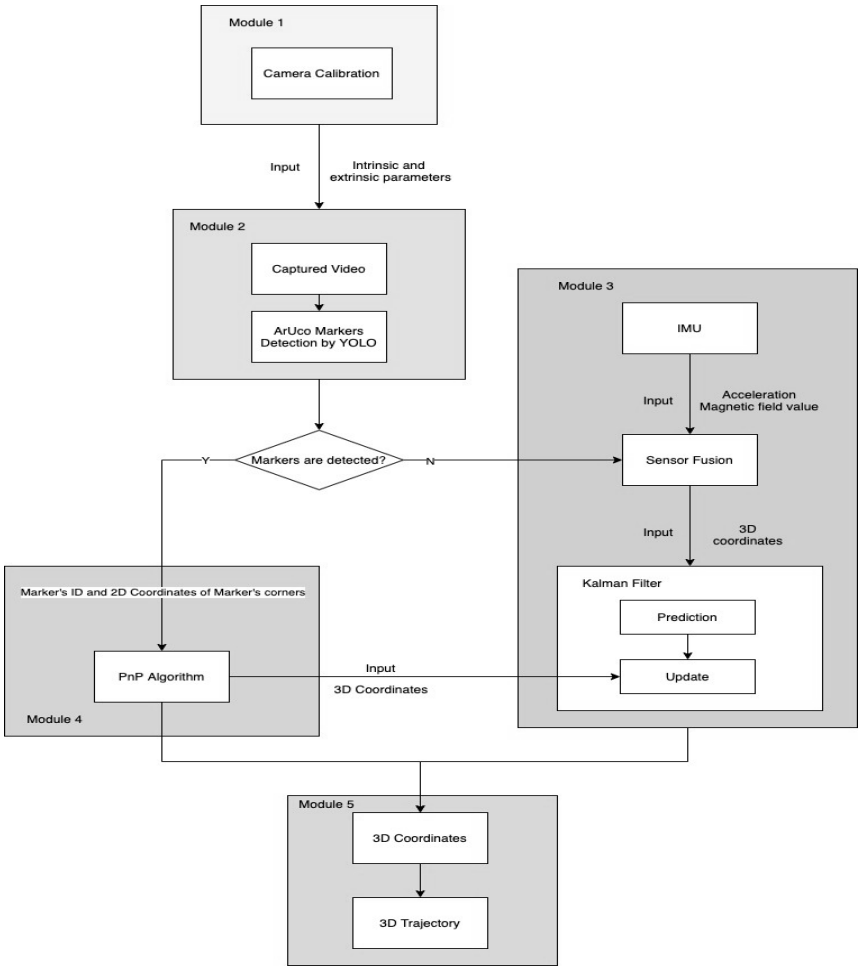


Figure. 1. System Design

2.2. Camera Parameter Estimation

Camera calibration is an essential initial step in machine vision applications. Its purpose is to establish a geometric model of camera imaging, enabling the determination of the 3D spatial position of a point on an object's surface and its corresponding location in the captured image. The parameters defining this geometric model are known as the camera parameters. Obtaining these parameters typically involves a combination of experimental measurements and calculations. This iterative process, involving the determination of intrinsic, extrinsic, and distortion parameters, is commonly referred to as camera calibration [13].

Converting the world coordinate system to the camera coordinate system is the initial step in camera calibration. The world coordinate system (x_W, y_w, z_w), which is also referred to as the measurement coordinate system, enables the description of the spatial position of both the camera and the object being measured. The placement of the world coordinate system can be flexibly determined based on the specific circumstances. On the other hand, the camera coordinate system (x_c, y_c, z_c) is also a three-dimensional right-angle coordinate system, with its origin located at the optical center of the lens. The x and y axes run parallel to the sides of the image plane, while the z-axis represents the lens optical axis and is perpendicular to the image plane. Equation (1) illustrates the process of conversion.

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{1}$$

in the homogeneous matrix $\begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}$, $\boldsymbol{R}$ represents the rotation matrix (3×3), $\boldsymbol{t}$ represents the translation vector $(3 \times 1$ column vector), $(x_c, y_c, z_c, 1)^T$ and $(x_w, y_w, z_w, 1)^T$ are the homogeneous coordinates of an object in the camera coordinate system and the world coordinate system, respectively.

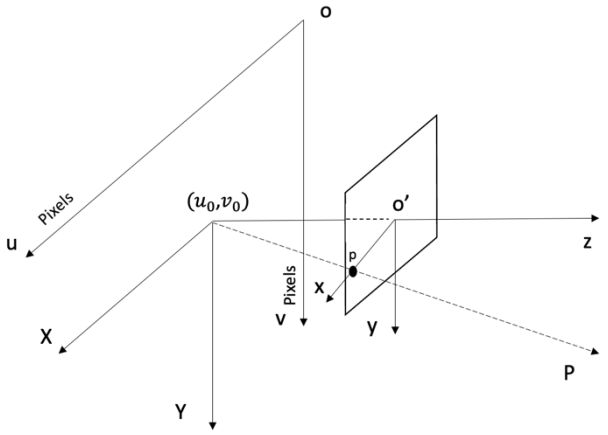


Figure. 2. The Pinhole Imaging Principle in Image Coordinate System and Pixel Coordinate System

The conversion of pixel coordinates and image coordinates is the subsequent step after determining the object's position in the camera's coordinate system. Figure 2 illustrates the pixel coordinate system, uov, which represents the arrangement of pixels on the camera chip. It is a two-dimensional right-angle coordinate system with its origin, o, located in the upper left corner of the image. The u and v axes align with the image plane's sides, respectively, and their units are pixels.

In the image coordinate system (XOY), the axes are typically measured in millimeters (mm), and the origin coincides with the intersection of the camera's optical axis and the phase plane, known as the principal point and positioned at the center of the image. The X-axis and Y-axis of the image coordinate system are parallel to the u-axis and v-axis, respectively. Therefore, the relationship between these two coordinate systems is translational,

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{dX} & 0 & u_0 \\ 0 & \frac{1}{dY} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{2}$$

where dX and dY represent the physical dimensions of a pixel in the X and Y-axis directions, respectively, and u_0 and v_0 correspond to the coordinates of the principal point.

Perspective projection, as demonstrated by the matrix below, involves the relationship between a point P (with coordinates in the camera's local reference frame $[x_c, y_c, z_c]^T$) in 3D space and its corresponding image point p, as depicted in Figure 2. The line connecting P and the optical center o' of the camera is referred to as $o'P$. The projection of point P onto the image plane p occurs at the intersection of $o'P$ and the image plane.

$$s \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \tag{3}$$

where the effective focal length, f, represents the distance from the optical center to the image plane, and s represents the nonzero scale factor. $(x_c, y_c, z_c, 1)^T$ represents the camera local reference frame (xoy) homogeneous coordinates of the 3D point P, and $(X, Y, 1)^T$ represents the image coordinate system (XOY) homogeneous coordinates of the image point p.

Therefore, the equation below serves as a representation of the geometric model of camera imaging, encompassing the camera's intrinsic and extrinsic parameters,

$$s \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = M_1 M_2 X_w \tag{4}$$

where, the intrinsic parameter is $\boldsymbol{M}_1 = \begin{bmatrix} a_x & 0 & u_0 & 0 \\ 0 & a_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$ with , $a_y = \frac{f}{dY}$, and the extrinsic parameter is $M_2 = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}$.

2.3. Monocular Vison Detection

The principle of vision-based detection was implemented in Module 2. As a single-stage detector with excellent performance in terms of processing efficiency, YOLO meets the required real-time performance of the proposed system [14][15]. For object detection, YOLO automatically resizes the image into small square grid with side lengths of 52, 26, and 13 to segment small, medium, and large targets, respectively, for different sizes. However, at initial input, the image is first resized to a 416*416 grid. This determines the lower limit of the object size and the upper limit of the detection distance. Each smaller grid will be responsible for the detected objects it contains.

YOLO Training. We train the YOLOV3 on Google Colab. The framework we use is Darknet, which we use to train our custom dataset. In the first step, we need to modify the file that defines the network structure according to our self-define dataset. Since our dataset contains only one category, which is the ArUco code, the corresponding number of filters can be calculated as 18 according to the formula below.

$$filters = (classes + 5) * 3$$

where filters and classes indicate the number of filters and classes. After uploading the dataset and the corresponding files to Google Colab, we can start training the model. A weight file is obtained at the end of training and will be used in the process of detecting markers.

Markers Detection in Android. After training the neural network, our task becomes to make the neural networks detect markers on the phone. We use OpenCV to help with this task. First, we need to add OpenCV as a dependency of the project. This step was done in the previous camera calibration. Place the weight file obtained after training into the Android project and load the weight

file with the function getAssetsFile (). The next step is to pass the camera frame into the network for detection. Finally, the coordinate points of the four corners of the detected markers are returned.

Detecting Test. We use 200 images in the dataset, as a test set to validate the detection performance of YOLOV3. The test set has 200 files and contains 784 ArUco code markers. Then we use the YOLOV3 to detect the test set. The detection results show that out of 784 objects to be detected, 368 were detected incorrectly.

Evaluation Metrics. Based on the above detection results, we can then evaluate the neural network model we have trained. We use mean Average Precision(mAP) to evaluate our training result. Before that, we need to introduce
a few metrics.

1) *Precision*. Precision measures how accurate is model's predictions. i.e., the percentage of the model's prediction is correct

$$Precision = \frac{TruePostive}{TruePostive + FalsePositive}$$

2) *Recall*. Recall measures how a good model finds all the positives.

$$Recall = \frac{TruePostive}{TruePostive + FalseNegative}$$

3) IoU. IoU stands for intersection over unity. The overlap of two borders is measured by IoU. We use this to determine how many of the models' predicted boundaries overlap with the actual object boundary.

$$IoU = \frac{AreaOfOverlap}{AreaOfUnion}$$

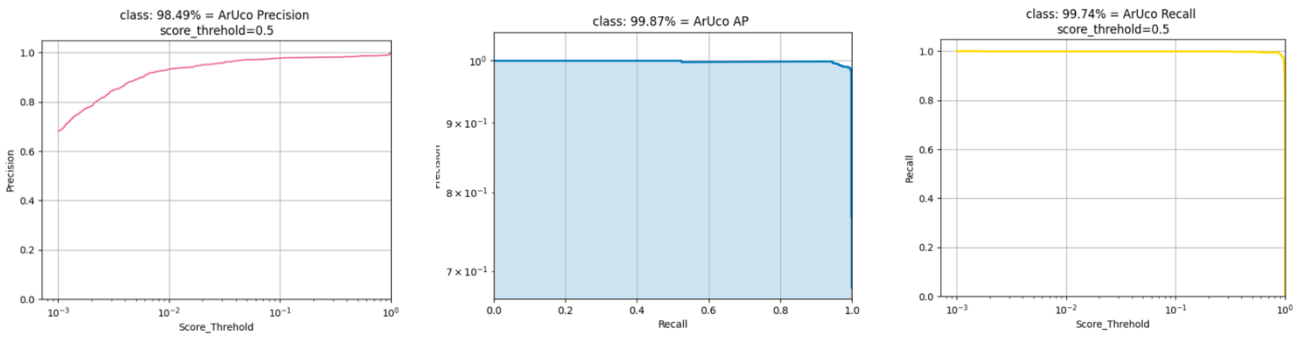
(

4) *AP (Average precision*). AP calculates the average precision value for recall values ranging from 0 to 1. The area under the precision-recall curve is the general definition for the AP. Precision and recall are always in the range of 0 to 1. As a result, AP is also between 0 and 1.

$$AP = \int_0^1 p(r)dr$$

5) *mAp.* The mAp is the average of AP. Since we only have one class which is the ArUco code, so mAp is equal to AP.

Result. Figures 3 show the precision, recall, and mAP of our trained model, respectively. As can be seen from the results, our model has achieved a 99.87% mAP. This detection accuracy is high and satisfies our requirements of marker detection for indoor localization.



a) Precision *b) Average Precision* *c) Recall*

Figure.3. The Performance of Our Trained Model

2.4. Camera Pose Estimation

The unique ID and 2D coordinates of the subject obtained from the visually tracked video frame will be used as input for the pose estimation of the mobile device. The pose estimation of the device mainly makes use of the Perspective n Point algorithm. The usual algorithms are broadly classified into iterative and non-iterative types. n general conditions exist for the PnP algorithm [16], i.e., there are n 3D reference points in the 3D earth coordinate system and these reference points can be projected on the captured image.

The full name of the EPnP algorithm is Efficient Perspective-n-Point, a scheme that expresses the camera coordinates of the reference points as a weighted sum of the control points, and then transforms the problem into a solution to the camera coordinate system of these four control points [17]. For the non-planar case, four non-coplanar control points are required, whereas for the planar case, only three are required. In contrast, in most PnP algorithms, the depth of the feature points is often the first object to be solved. Barycentric points will be used to calculate the final center coordinates to obtain the specific tilt angle. The process of this algorithm and its comparison with the conventional PnP algorithm is discussed in detail in 2.7 of this chapter. It is worth stating that EPnP meets the requirements of the new system as an algorithm belonging to a relatively fast processing speed. This can also be demonstrated in the subsequent experiments.

2.5. Sensor Fusion

Role of Inertial Measurement Unit (IMU). The IMU is a sensor that monitors the acceleration and angular velocity of each axis of the 3D coordinate system. It mainly consists of an accelerometer and a gyroscope. The former provides linear acceleration in all directions of the coordinate system. The latter outputs angular velocity. It monitors acceleration, angular velocity, and magnetic field and estimates motion, direction, and heading by interfacing with sensor fusion software [18][19]. It is worth mentioning that sensor fusion can be done by relying on the smartphone's original firmware. In the newly designed system, we have focused on providing a solution to the situation where visual detection is temporarily "blinded" in adverse environmental conditions. In this case, sensor fusion is proposed as a supplementary solution. This physical detection, which does not rely on camera or electronic platform processing, will be an important part of the new system.

Coordinate System Conversion. Converting the coordinate system of an Android phone's acceleration sensor is essential for accurate acceleration measurements. To achieve this, we must transform the phone's coordinate system into an inertial, non-rotating coordinate system known as the Earth coordinate system. By performing this conversion, we enable the Android phone to accurately measure acceleration vectors regardless of its orientation, allowing for precise calculation of the phone's trajectory within the Earth coordinate system. The transformation from the phone's coordinate system to the Earth's coordinate system, as depicted in the formula below [20], accomplishes this conversion.

$$\boldsymbol{a}_w = \boldsymbol{R}_z(\psi)\boldsymbol{R}_y(\theta)\boldsymbol{R}_x(\phi)\boldsymbol{a}_l \tag{5}$$

where $\boldsymbol{a}_l = \left[a_l^x,\ a_l^y,\ a_l^z\right]^T$ represents the phone's linear accelerations in its local reference frame, $\boldsymbol{a}_w = \left[a_w^x,\ a_w^y,\ a_w^z\right]^T$ represents the phone's acceleration in the world reference frame, and $\boldsymbol{R}_z(\psi), \boldsymbol{R}_y(\theta), \boldsymbol{R}_x(\phi)$ represent the rotation matrices, and Euler angles ($\psi,\ \theta, \varphi$) correspond to the pitch, roll, and yaw as shown in Figure 4.

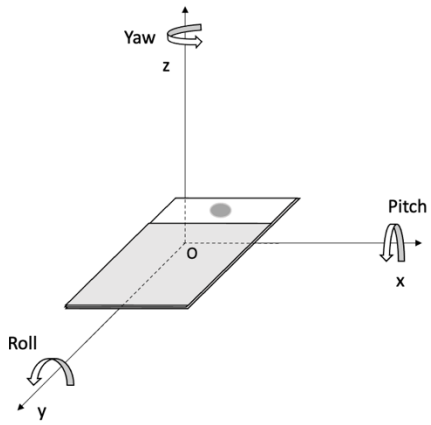


Figure.4. Mobile Phone Local Reference Frame

Due to the differences between the smartphone platform and the general coordinate system (Earth), we need to convert the 3D reference standard. Figure 5 shows the difference in the comparison of accelerometer readings after the coordinate system has been converted from the general to the mobile phone coordinate system.

According to Newton's laws, the acceleration of gravity is approximately 10m/s to the nearest single digit, while the horizontal direction should not have an acceleration in the absence of external forces. Here, the values on the X.Y.Z axes behave in accordance with Newton's laws as presumed.

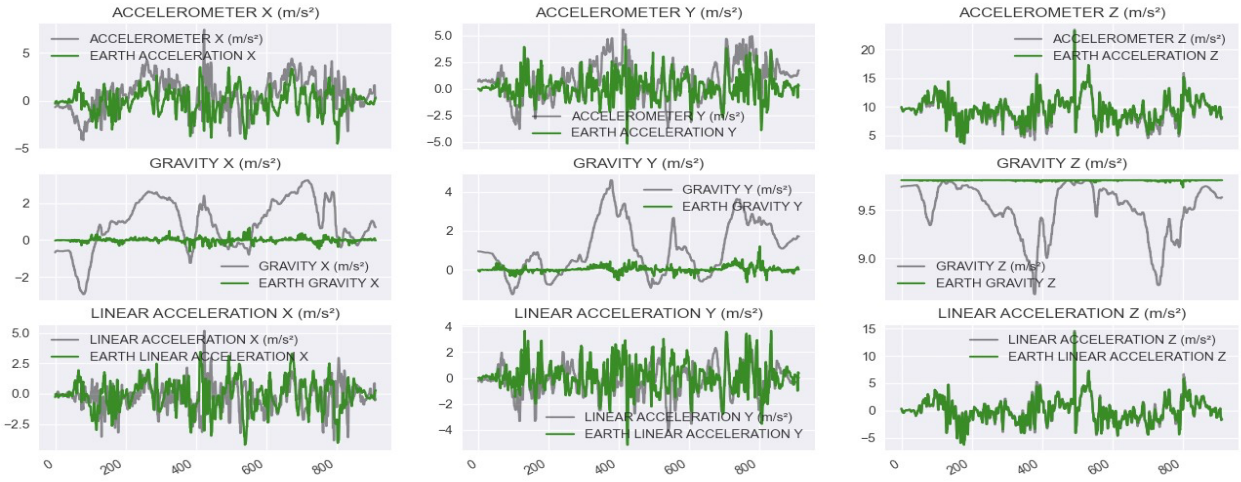


Figure. 5. Change in Sensor Readings Before and After 3D Reference System Transformation

(Grey for Pre-application and Green for Post-application)

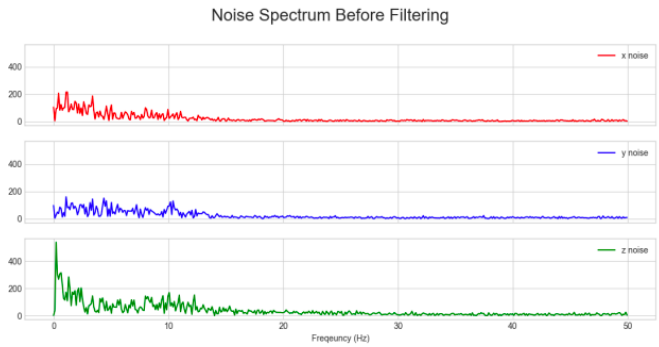


Figure.6. Noise Spectrum of Each Axis

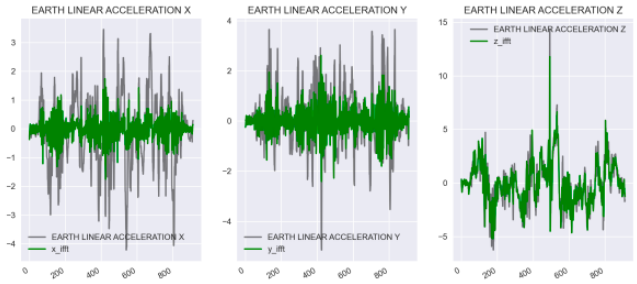


Figure.7. Accelerometer Values before and after the Low-pass Filter be Applied.
(Grey for Pre-application and Green for Post-application)

Noise Filtering: To estimate the distance traveled each time, the acceleration needs to be integrated twice[21]. It is important to note that the integration of the noise also needs to be considered. Its value has a potential impact on the error of the results in the 3D coordinate system. However, we are not sure what type of filter needs to be used at this point. Therefore, as an attempt a low-pass filter was introduced to attenuate the noise. The accelerometer readings were subjected to Fourier analysis to obtain the spectrum of the noise. Figure 6 shows the noise spectrum in each direction of the coordinate system. It shows that the noise has been concentrated in the low-frequency band, and it looks like we should use a high-pass filter to be the right choice. However, the signal we are supposed to measure is also concentrated in the low-frequency band, so a low-pass filter is a genuinely right choice. Using a lowpass filter can make the signal smoother. Figure 8 shows the variation of the acceleration values after the low-pass filter. An acquisition frequency of 100 Hz was used in the test of Figure 7. The x-axis is the IMU reading, and the y-axis is the number of samples taken. As can be seen from Figure 7, the values on each axis are attenuated, which corresponds to a partial attenuation of the noise.

2.6. Motion Tracking Correction

When sensor fusion is activated as an auxiliary system, the primary sensors such as the accelerometer and gyroscope are the first to start operating and providing raw readings. Errors in the raw readings due to the sensors themselves may be accumulated after two integrations of the measured distance. Therefore, we introduce Kalman filtering [22] to correct for this. It is divided into two main parts. The first part is the prediction, which refers to the extrapolation of the current stage value from the raw readings of the previous stage. The second part is updating, which refers to linear iteration to estimate the optimal solution based on the obtained values of the current stage combined with the values of the previous stage [23]. This process is repeated at the next stage to update and correct. As the proposed system seeks a certain processing speed, this approach, which does not need multiple parameters but only process quantities, excellently matches the needs of the new system.

The process of constructing the Kalman filter state space for this project is described in detail below. The position and velocity of the device at moment t can be deduced from the position and velocity at moment $t - 1$.

$$\boldsymbol{P}_t = \boldsymbol{P}_{t-1} + \boldsymbol{V}_{t-1} \times \delta t + \frac{\delta t^2}{2}\boldsymbol{a}_{wt} \tag{6}$$

$$\boldsymbol{V}_t = \boldsymbol{V}_{t-1} + \delta t \times \boldsymbol{a}_{wt} \tag{7}$$

Where P, V, a, and δt, are position, velocity, acceleration, and time interval, respectively. Combining equations (6) and (7), in a three-dimensional coordinate system, we can obtain,

$$\widehat{\boldsymbol{X}}_t^- = \boldsymbol{A}\widehat{\boldsymbol{X}}_t^- + \boldsymbol{B}\boldsymbol{a}_t \tag{8}$$

Where A is the state transition matrix, B is the control matrix, a_t is the acceleration at moment t.

$$\widehat{\boldsymbol{X}}_t^- = \begin{bmatrix} P_t^x \\ P_t^y \\ P_t^z \\ V_t^x \\ V_t^y \\ V_t^z \end{bmatrix} \quad \boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 & \delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{B} = \begin{bmatrix} \frac{\delta t^2}{2} & 0 & 0 \\ 0 & \frac{\delta t^2}{2} & 0 \\ 0 & 0 & \frac{\delta t^2}{2} \\ \delta t & 0 & 0 \\ 0 & \delta t & 0 \\ 0 & 0 & \delta t \end{bmatrix} \quad \boldsymbol{a}_t = \begin{bmatrix} a_t^x \\ a_t^y \\ a_t^z \end{bmatrix}$$

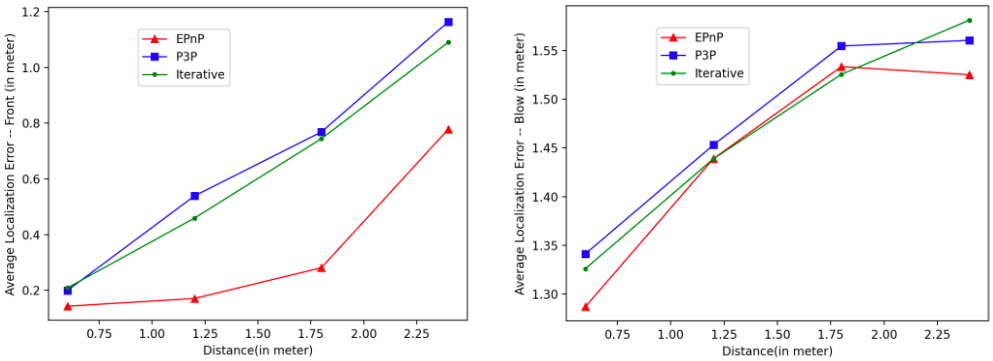
Thus, the measurement matrix is

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

2.7. EPnP Algorithm

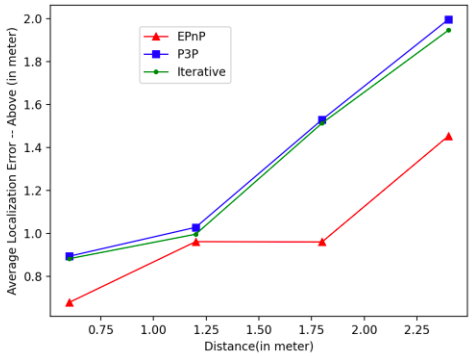
We used PnP's coordinate system conversion method during the camera calibration section. In this section will further explain its role and verify the superiority of EPnP in the experiments.

EPnp, or Efficient Perspective n Point, is an optimization solution for estimating camera angles. It uses a non-iterative approach to estimate the camera pose from a correspondence of n 3D to 2D points. [24]. We need to verify that the EPnP algorithm is superior to other PnP algorithms. Therefore, we designed this experiment to compare the difference in accuracy and speed of computation between EPnP, P3P, and Iterative algorithms. In this experiment, the origin of our world coordinate system is the center of the marker, with the X-axis pointing to the right, the Y-axis pointing up, and the Z-axis facing outward perpendicular to the wall. We took pictures of different markers from different angles (Front, below, above, left, and right side) and different distances (0.6, 1.2, 1.8, 2.4 meters), and noted the camera's position at the time the picture was taken. Then, we use three different methods -EPNP, P3P, and Iterative - to calculate the camera position in world coordinates. Therefore, the camera had a total of 20 different positions, and we took 30 images for each of these different positions and then performed thirty experiments for each position to calculate the average error. Figure 8 shows the error comparison of different PnP algorithms for calculating mobile devices at different angles and different distances.

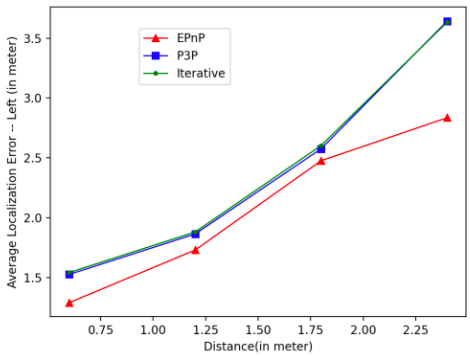


a) Front

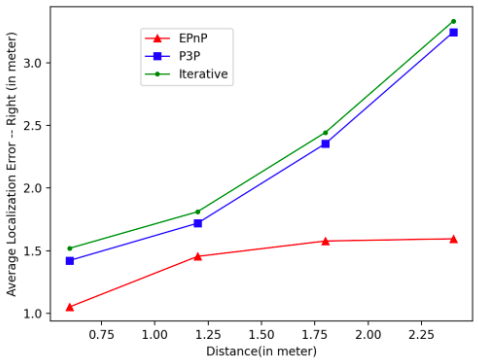
b) Below



c) Above



d) Left



***e)** Right*

Figure.8. Comparison of the Different Algorithm's Average Localization Error in Different Angles

Figure 8a) compares the average error of different algorithms for estimating the camera position at different distances when the mobile device is located directly in front of the marker. The Iterative and P3P algorithms perform similarly, with the EPnP algorithm showing a clear advantage. The errors of all three algorithms increase with increasing distance. Figure 8b) shows the performance of the different algorithms when the camera is located directly below the marker. The error of the three algorithms still increases with distance, and the performance of the three algorithms is very similar. However, the EPnP algorithm still has a slight advantage. Figures 8c) and 8d) show the comparison of the errors in estimating the camera position by different algorithms when the marker is located directly above the camera and to the left of the marker, respectively. The errors of the P3P and Iterative algorithms are very similar when estimating these two different angles, and the error curves almost overlap. Again, the computed error is positively correlated with the distance. EPnP algorithm has an obvious advantage over the other two algorithms. Figure 8e) compares the performance of different algorithms when the camera is on the right side of the marker. The difference between here and the previous one is that the EPnP algorithm is not significantly affected by the distance.

Table I. Average Performance Comparison of Different PnP Algorithm

Algorithm	Time (ms)
EPnP	223.658
P3P	657.869
Iteration	1110.194

As can be seen from these experimental results, the computational error of each algorithm increases with the distance. The reason is that when the marker is farther away from the camera, the smaller imaging area on the image. The x, and y coordinate information in the physical world becomes a minimal number of pixels in the image. This causes an error in the z-axis position. The closer the marker is to the camera, the larger the imaging area and the greater the accuracy. In addition, the lighting condition at the time the photo was taken can also affect the calculation results. In general, the EPnP algorithm has the slightest error and the superiority of the EPnP algorithm is better demonstrated when the distance is farther. Besides, we also performed performance tests on the three algorithms. We executed the same program 1000 times with each of the three algorithms to obtain the running time of the algorithms. Table I shows the results of the three different algorithms. From the results, the EPnP algorithm is the fastest, followed by the P3P algorithm. The iterative method is the slowest.

2.8. Dataset

Figure.9. Dataset Structure

Data Collection. In this project, we need to collect our own dataset because there is no open-source dataset available. The dataset usually consists of a training set and a test set. In general, the total number of training sets is greater than the total number of test sets.

Due to the fast movement of the camera, some problems may occur in the test set, such as blurred images caused by the fast movement of the camera. Therefore, the dataset should also include images with blurred markers. To meet these requirements, we first took many images and then further processed them. We used a combination of Python and OpenCV to add image effects.

Data Tagging. In machine learning, data labeling is defined as a form of data processing that enables information marking and information recognition of the original data [25]. Specifically, the tracking of a specific object in a specific image and the filtering of audio or video content features can be scenarios in which it can be put to great use. A tag in a neural network is a visual identifier that uses Qt as a graphical intervention [26].

Labeling [27]-[29] supports the YOLO format. For each image file in the same directory, a file with the same name is created in the YOLO dataset format. Each file comprises the picture file that corresponds to the object class, object coordinates, height, and width. The result of the labeling can be saved as an XML file or as a TXT file. This labeling process requires us to manually mark the boxes that surround the markers. After drawing a box each time, we must mark what object the box belongs to. After finishing the labeling of one image, we can import the following images and cycle through them. Images are classified according to their different conditions, such as illumination and blur. Four-fifths of the total data set is used for training and one-fifth for testing. As can be seen from Figure 9, this dataset does not have a validation set. The reason is that the validation set is automatically divided from the training set at the beginning of the model training. The test set is needed to evaluate the model performance. Figure 9 illustrates the data set structure.

3. Experimental Performance Testing

The theoretical logic of the proposed system needs to be supported by practical experiments. In order to quantify the innovative performance of the new system compared to traditional vision tracking systems or sensor positioning systems, a series of experiments were designed to quantify the performance and verify the effectiveness of the improvements.

First, we divided the experimental groups into three categories: vision-based localization systems, sensor fusion-based localization systems, and the proposed new system. There are 15 experimental groups for each type of experiment. The results of the experiments will be subjected to data analysis.

The location where this experiment will be conducted is the Prairie Springs Science Centre at University. The tools to be used include a Nokia 7.2 as the intelligent example platform for the experiments and a tape measure. The ArUco code was measured accurately and attached to the walls of the laboratory at different heights and orientations. A corner of the laboratory is shown in Figure 10 as an example. During the experiment, a partially illuminated, shaky, and noisy environment was deliberately designed to test the impact of several indicators of the new system and to assess its resistance to interference.

Figure 11 shows a top view of the experimental environment. The experiment will follow the route shown in Figure 11. The six yellow dots as selected key points will be used in subsequent test sections for examples of data collection. In addition, we have introduced six high-frequency depth cameras which will cover the experimental area and ensure the relative accuracy of the real trajectory.

Figure.10. The Environment of the Laboratory

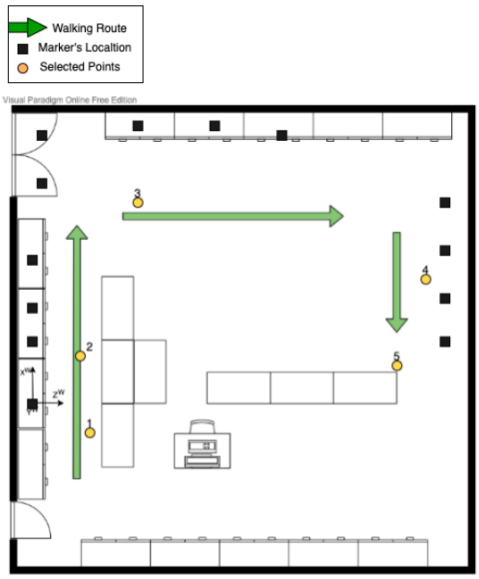


Figure.11. The Walking Route Design for the Experiment

Figure 12 shows the trajectory of the real movement made by tracing the points. Like the green arrows can be seen in Figure 11, the real path consists of three planes and two corners. Movement in the vertical direction is also included in the 3D trajectory. The maximum movement distance for horizontal movement is approximately 11.83 m, while the upper limit in the vertical direction is 6.53 m. At the same time, six high-frequency depth cameras were introduced, which will cover the experimental area and examine the 3D trajectory points to ensure the relative accuracy of the real trajectory.

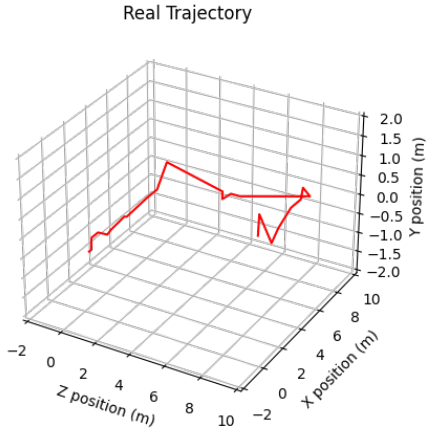
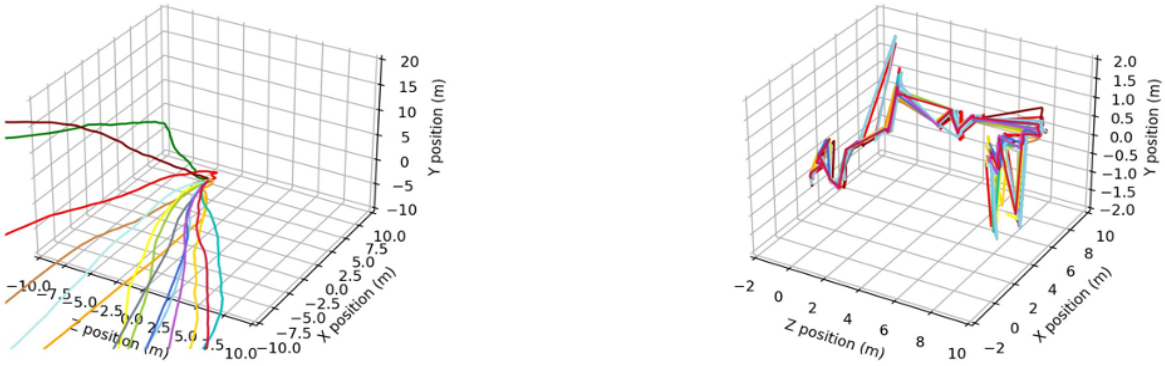
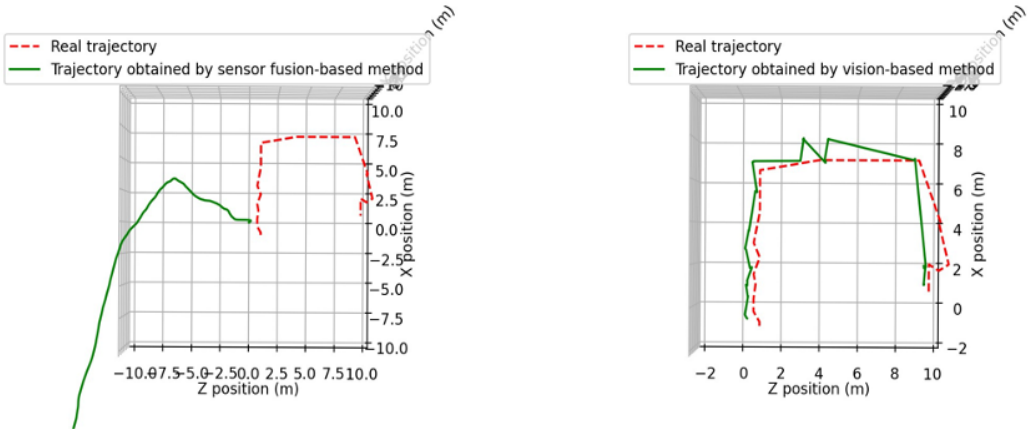


Figure.12. Actual 3D Moving Trajectory Drawing



a) Sensor Fusion-based b) Vision-based method

Fig. 13. Fifteen Trajectories Obtained by Sensor Fusion-based Method and Vision-based Method.



a) Sensor Fusion-based b) Vision-based

Fig. 14. Comparison of Real Trajectory and Trajectory Obtained by Sensor Fusion-based method and Vision-based Method.

Figure 13A and Figure 14A show the 3D trajectory and 2D trajectory of the sensor fusion system in fifteen sets of experiments (here in the Z-direction), respectively. The 2D trajectories are compared to the real trajectories to show the offset. In Figure 13A the 3D trajectories are shown in different colors for different groups of experiments. We can see that they clearly diverge from the original trajectory as the distance from the position increases. This reflects the instability of the system under this approach. A comparison with the real trajectory in Fig. 14A shows a large discrepancy, with relatively large shifts in direction and position. The reason for this phenomenon is mainly due to the structure of the IMU itself, which has a certain amount of unavoidable drift during the testing process. Further, the sensor has non-zero readings due to the inertia of the moving object as it returns to rest from movement. In addition, the presence of gravitational acceleration in the vertical direction will also have an effect on the sensor readings, which cannot be completely avoided. The use of double integration when calculating distances adds to this error.

Figure 13B and Figure 14B show the trajectory performance of a conventional vision tracking system in 3D and 2D (Z-direction for example) respectively. Again, its 2D trajectory is compared with the real trajectory. In the 3D trajectory in Figure 13B, the different groups of experiments are indicated by different colors. Each group of trajectories does not appear to deviate excessively in general and follows a relatively consistent direction. Figure 14B shows that the trajectories (green) have some degree of fluctuation and bulging. This is because visual capture failed in some experimental groups with poor environmental factors, which resulted in marker positions not being accurately estimated. Therefore, until the next valid visual point is captured, the position will be transiently shifted to the existing point. This is also the reason why some of the unexpected fluctuations in the figure were generated. This reflects at a deeper level the weak resistance of the original system to disturbances, especially when the visual situation is poor, which is exactly what the optimized system needs to achieve.

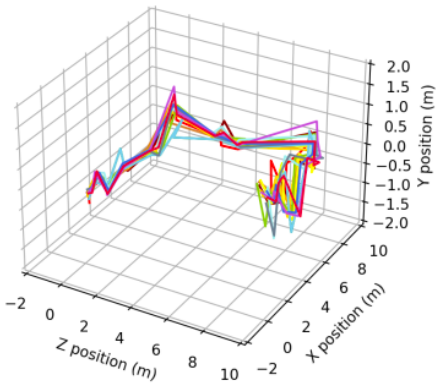


Fig.15. 3D Trajectory Obtained by our System

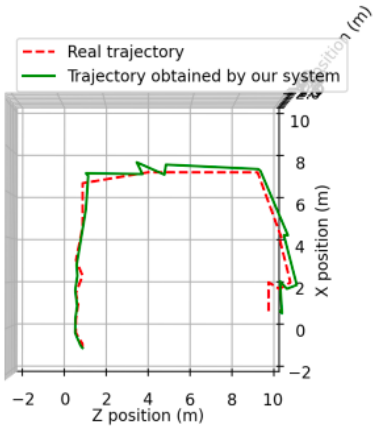


Fig.16. Comparison of Proposed System Trajectory with Real Trajectory

The third part of the experiment was to evaluate our proposed method and test the reliability and validity of our system. Figure 15 shows the 3D trajectories for fifteen repetitions of the experiment in different conditions. Figure 16 points out that there are no large jumps in the trajectories obtained, which means that sensor-based localization successfully fills the gap when vision-based localization cannot be used. This largely suggests that this method combines the advantages of both methods and achieves optimization.

Figures 15 and 16 show the performance of the proposed method for both the 3D trajectory and the 2D trajectory (in terms of Z position). By comparing the 3D trajectory in Figure 15 with Figure 13 it can be seen that the trajectory is much tighter for the different experiment sets in Figure 15. This implies an improvement in the stability of the system. The disappearance of gaps and jumps indicates that the gaps in its visual detection are effectively filled by the sensor. In terms of qualitative analysis, these signs become relatively strong evidence of the success of the optimization.

The next step in the experiment is to quantify the scale of this optimization to comprehensively measure the extent of this improvement.

Table II. Localization Results by Using Different Methods

No.	Real	Proposed System	Vision-based	Sensor-based
1	-1	-1.4011	-1.6379	-8.9199
	-0.15	-0.5655	1.0470	-2.2626
	1	-1.3943	1.3849	-3.8799
2	1	1.3202	2.0098	-6.4524
	-0.2	-0.5938	1.6148	-2.3476
	0.8	1.1437	1.3625	-2.8963
3	7	7.3121	7.8599	0.3548
	0	0.4543	1.1423	-1.3658
	4	4.1368	5.2847	1.8462
4	4	4.4975	2.9463	-3.4625
	1.2	1.6831	2.2486	-2.3746
	10	10.4304	10.2109	0.4625
5	0.6	1.1250	2.1186	-7.3624
	1	1.3554	2.2250	-2.4762
	9.5	10.3051	10.1139	5.7264

Table III. Localization Errors Comparison

No.	Proposed System	Vision-based	Sensor-based	Impr1	Impr2

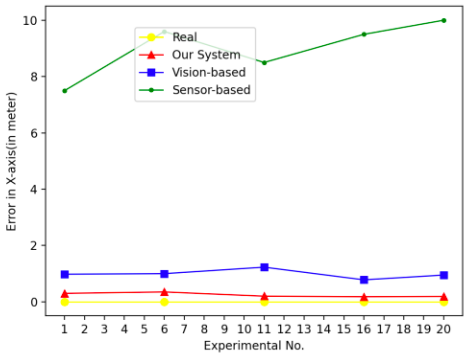
1	0.4011	1.3621	7.9199	71%	95%
	0.4155	1.1030	2.1126	69%	80%
	0.4057	1.6159	4.8799	75%	92%
2	0.4798	1.0980	7.4524	56%	94%
	0.4062	1.8148	2.1476	78%	81%
	0.4675	1.4375	3.6963	67%	87%
3	0.4897	1.1401	6.6452	57%	93%
	0.4543	1.1423	1.3658	60%	66%
	0.6632	1.2847	2.1574	48%	69%
4	0.4975	3.0537	7.4625	83%	93%
	0.4831	1.0486	3.5746	54%	86%
	0.4304	1.7810	9.5375	76%	95%
5	0.5250	1.5186	7.9624	65%	93%
	0.4446	1.2250	3.4762	64%	87%
	0.8051	1.3864	3.7736	42%	79%

** Units in meters*

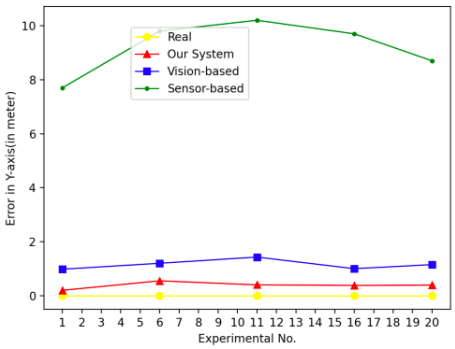
Table II shows the example of location results evaluated at the selected positions (as indicated in Figure 11). The yellow dots in Figure 11 represent these selected points. The location error comparison of our proposed method, vision-based method, and sensor fusion-based method is shown in Table III.

Through observation, the experiments revealed that the error range of the sensor fusion system fluctuated between 1.36-9.53m, and the error value of the conventional vision-based localization system fluctuated between 1.04-3.05m but was always greater than the proposed system. The proposed approach outperforms the competition in terms of tracking precision, with a mean error calculated to be 0.4912 meters approximately.

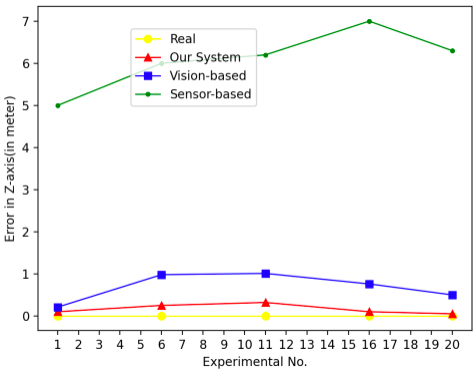
On average, the proposed system improves by approximately 66% on the X-axis, 65% on the Y-axis, and 62% on the Z-axis compared to systems with conventional vision tracking. In addition, our proposed method shows a notable improvement compared to sensor-based methods. The improvement is about 94% accuracy on the X-axis, about 80% on the Y-axis, and 84% on the Z-axis. The tracking errors of these fifteen experiments on the three axes are demonstrated in Figure 17a), figure 17b), and Figure 17c), respective to X, Y, and Z axes.



a) X-axis.



b) Y-axis



c) Z-axis

Figure.17. Errors in the 3D-coordinate system for Different Localization Methods

By comparing the three previous trajectory maps, the experiments demonstrate a closer fit of the proposed system to the original trajectory, which differs from the relatively large gap based on traditional vision-based fluctuations and sensor fusion. this is a linear qualitative analysis. By quantifying the tabular data for each of the 15 test groups of the approach, it was found that the maximum error of the new system was 0.8051m, while the mean was approximately 0.4912 meters, a visible improvement for both the sensor fusion-based localization system with an average error of 4.94426 meters and the conventional vision-based tracking system with an average error of 1.467 meters. This is reflected in all three axes. Interestingly, the experiments also revealed that the system seemed to perform unusually in the Y direction compared to the other axes. Additional tests were conducted to explore this near-arc trajectory and two possibilities were speculated: the first was that the effect of gravitational acceleration on the sensor could not be eliminated, which was brought about by the IMU; the second was due to the small relative distance of the vertical targets in the experiments, which caused an offset in the number of detections. But overall, it is undeniable that the new system has passed a series of tests on interference and poor environments, and that its strength and robustness have been relatively improved compared to the original system.

4. Conclusion And Future Direction

In summary, this work combines traditional monocular visual localization with sensor fusion. The experiments conducted point out that this combination complements the detection shortcomings when the visual environment is poor, which greatly enhances the environmental immunity of the new system. In this process, the integrated YOLO neural network performs marker detection and uses the ArUco code as a reference point, significantly reducing the probability of error through the collection of data sets. On the other hand, IMU localization and Kalman filtering for sensor fusion are integrated as part of the auxiliary system to fill the marker detection gap. The excellent prediction and updating capabilities of the Kalman filter improve the detection abilities of the proposed system. Our extensive experiments illustrate that a 64.5% improvement in tracking accuracy is achieved over conventional vision tracking and an 86% improvement is achieved over sensor detection. The average target processing speed of the new system is around 0.1638 seconds, which maintains the high speed and real-time nature of visual localization. The system's impressive tracking accuracy of 0.5m or less can be achieved without the need for expensive depth cameras or LIDAR technology, making it a cost-effective solution with potential applications for affordable indoor tracking methods, particularly on smartphone platforms.

The application prospects of the new system are expectable based on excellent performance in the test phase. There are still some optimization parts that can be completed in the future. On the one hand, diverse reference codes and new versions of the YOLO neural network may be used to compare and validate the most suitable solutions for the proposed new system. On the other hand, more extensive datasets will be collected to improve the labeling accuracy of the neural network. This will help to improve the overall stability and accuracy of the new system.

5. References

[1] Taha, Bilal, and Abdulhadi Shoufan. "Machine learning-based drone detection and classification: State-of-the-art in research." *IEEE access* 7 (2019): 138669-138682.

[2] Ahrens A, Lund KD, Marschall M, Dau T. Sound source localization with varying amount of visual information in virtual reality. PloS one. 2019 Mar 29;14(3): e0214603.

[3] Dostalek P, Vasek V, Kresalek V, Navratil M. Utilization of audio source localization in security systems. In43rd Annual 2009 International Carnahan Conference on Security Technology 2009 Oct 5 (pp. 305-311). IEEE.

[4] Lindner, T., Fritsch, L. Plank, K., and Rannenberg, K.: "Exploitation of public and private Wi-Fi coverage for new business models," in *Building the E-Service Society*. Springer, pp. 131–148, 2004.

[5] Violettas, G., TTheodorou, L., and Georgiadis, C.: "Netargus: A SNMP monitor & wi-fi positioning, 3-tier application suite," in *2009 Fifth International Conference on Wireless and Mobile Communications*. IEEE, pp. 346–351, 2009.

[6] Kunhoth, J., Karkar, A., Al-Maadeed, S., & Al-Ali, A., "Indoor positioning and wayfinding systems: a survey," *Human-centric Computing and Information Sciences*, vol. 10, pp. 1–41, 2020.

[7] Zheng X, Bao G, Fu R, Pahlavan K. The performance of simulated annealing algorithms for wi-fi localization using google indoor map. In2012 IEEE Vehicular Technology Conference (VTC Fall) 2012 Sep 3 (pp. 1-5). IEEE.

[8] Damiani, Maria Luisa, and Colette Cuijpers. "Privacy challenges in third-party location services." In *2013 IEEE 14th International Conference on Mobile Data Management*, vol. 2, pp. 63-66. IEEE, 2013.

[9] Piasco, N., Sidibe, D., Demonceaux, C., and Gouet-Brunet, V.: "A survey´ on visual-based localization: On the benefit of heterogeneous data," *Pattern Recognition*, vol. 74, pp. 90–109, 2018.

[10] Alkendi Y, Seneviratne L, Zweiri Y. State of the art in vision-based localization techniques for autonomous navigation systems. IEEE Access. 2021 May 21; 9:76847-74.

[11] Lu Z, Liu F, Lin X. Vision-based localization methods under GPS-denied conditions. arXiv preprint arXiv:2211.11988. 2022 Nov 22.

[12] Oščádal P, Heczko D, Vysocký A, Mlotek J, Novák P, Virgala I, Sukop M, Bobovský Z. Improved pose estimation of aruco tags using a novel 3d placement strategy. Sensors. 2020 Aug 26;20(17):4825.

[13] Zhang, Z., "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[14] Redmon J, Farhadi A. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767. 2018 Apr 8.

[15] Zhao L, Li S. Object detection algorithm based on improved YOLOv3. Electronics. 2020 Mar 24;9(3):537.

[16] Pritsker, A., *Introduction to Simulation and SLAM II*. Halsted Press, 1984.

[17] Lepetit, V., Moreno-Noguer, F., and Fua, P.: "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, p. 155, 2009.

[18] Sabatelli S, Sechi F, Fanucci L, Rocchi A. A sensor fusion algorithm for an integrated angular position estimation with inertial measurement units. In2011 Design, Automation & Test in Europe 2011 Mar 14 (pp. 1-4). IEEE.

[19] Malyavej V, Kumkeaw W, Aorpimai M. Indoor robot localization by RSSI/IMU sensor fusion. In2013 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications, and Information Technology 2013 May 15 (pp. 1-6). IEEE.

[20] Spong, Mark W., Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*. Vol. 3. New York: Wiley, 2006.

[21] Kam, Ho Chuen, Ying Kin Yu, and Kin Hong Wong. "An improvement on aruco marker for pose tracking using Kalman filter." In *2018 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pp. 65-69. IEEE, 2018.

[22] Kalman, R. E., "A new approach to linear filtering and prediction problems" Journal of Basic Engineering vol. 82 no. 1 pp. 35-45 1960

[23] Simon, D., *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[24] Lepetit V, Moreno-Noguer F, Fua P. EP n P: An accurate O (n) solution to the PnP problem. International journal of computer vision. 2009 Feb; 81:155-66.

[25] Sun Y, Lank E, Terry M. Label-and-learn: Visualizing the likelihood of machine learning classifier's success during data labeling. InProceedings of the 22nd International Conference on Intelligent User Interfaces 2017 Mar 7 (pp. 523-534).

[26] Zhang S, Jafari O, Nagarkar P. A survey on machine learning techniques for auto labeling of video, audio, and text data. arXiv preprint arXiv:2109.03784. 2021 Sep 8.

[27] Huang R, Gu J, Sun X, Hou Y, Uddin S. A rapid recognition method for electronic components based on the improved YOLO-V3 network. Electronics. 2019 Jul 25;8(8):825.
[28] Wang D, Shang Y. A new active labeling method for deep learning. In2014 International joint conference on neural networks (IJCNN) 2014 Jul 6 (pp. 112-119). IEEE.
[29] Tzutalin L. Git code. Available from: https://github. com/tzutalin/labelImg 2015. [Accessed 31 May 2023].