Proceedings of the 7th RSI
International Conference on Robotics and Mechatronics (ICRoM 2019)
November 20-21, 2019, Tehran, Iran

ARAS-IREF: An Open-Source Low-Cost Framework for Pose Estimation

H. Damirchi, R. Khorrambakht, H. D. Taghirad, Senior Member, IEEE
Advanced Robotics and Automated Systems, Faculty of Electrical and Computer Engineering,
K.N. Toosi University of Technology, P.O. Box 16315-1355, Tehran, lran.
Email: hdamirchi, r.khorrambakht@email.kntu.ac.ir, taghirad@kntu.ac.ir

***Abstract*—Despite the amount of research reported on state estimation and sensor fusion in the field of robotics, there are no well known low-cost solutions for a referencing system to determine the accuracy of developed methods by providing a suitable ground truth for them. In this paper an efficient and accurate 6–DoF pose measurement system is proposed and implemented on a spherical parallel robot using IR LEDs. This approach uses the perspective-n-point algorithm to derive the transformation matrix representing the accurate relative pose of the end-effector with respect to an inertial frame. Exploiting a visible light filter in front of the camera has rendered this approach robust against illumination changes. Furthermore, it allows for mitigating the rolling shutter effects by reducing the exposure time. Finally, a custom made testing module is proposed to verify the accuracy of the proposed device, and the calibration process proves the accuracy and efficiency of the system.**

***Index Terms*—Referencing, pose-estimation, perception, robotics, orientation, calibration.**

I. Introduction

When reporting the efficiency of a newly developed state estimator or controller, a ground truth is needed to compare the accuracy and integrity of the estimation results. This ground truth is conventionally achieved either through perception methods, which are computationally expensive, or by using high-end special equipment that might not be available to every researcher.

Various approaches towards the usage of cameras for object tracking and pose estimation, [1]–[5] may be categorized into two main groups, namely, marker-based and perception based systems. In marker-based tracking systems, certain markers such as IR LEDs or tags, are attached to the target object or placed on a surface with known coordinates in the global frame. Using the correspondence between every specific LED in the image plane and the marker's local coordinate, one can find the camera's orientation and translation with respect to the marker's frame.

IR based tracking systems are limited to indoor motion capture and augmented reality applications since in outdoor environments the sunlight can affect the IR reception by the cameras. Hence, for outdoor applications, employing tags is a more common approach. On the other hand, the second category exploits the camera geometry and the correspondences between salient features in consecutive frames in order to estimate the camera's ego-motion. In fact, these systems solve the Structure from Motion (SfM) problem in a real-time manner. However, these systems are computationally expensive and the achievable accuracy using them is lower than that of marker based frameworks.

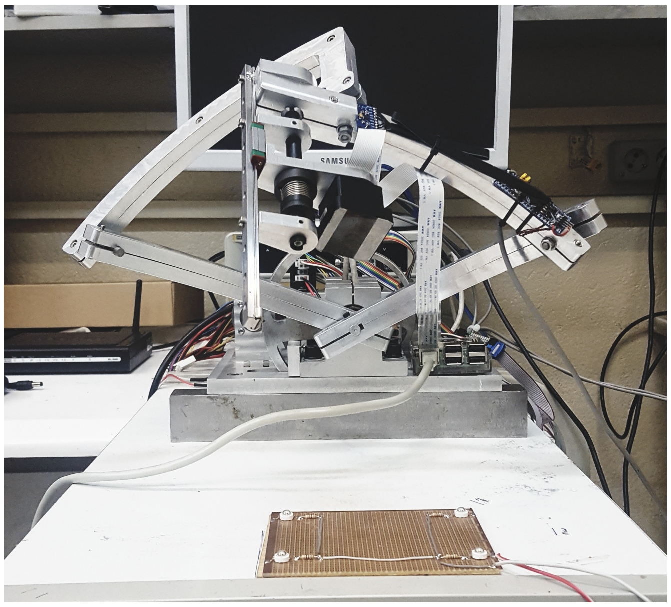
Fig. 1. The ARAS-DIAMOND Robot [6]

In this paper, by using multiple IR LEDs and a single camera, a pose estimation method is proposed that could be used as a referencing system for a wide range of robotic applications. These LEDs are scattered across a rectangular plane with known dimensions in the global frame and the camera is mounted on the robot's end-effector. The particular robot we used for implementation, is a spherical parallel manipulator called ARAS-DIAMOND [6]. One advantage of the proposed system originates from employing visible light filters in front of the camera, which makes the system robust against illumination changes and noises that are introduced from cluttered environments. Furthermore, since the retrieved frame to be processed is a clean image containing only the important features, the required image processing and feature detection subsystems are simpler and more efficient. Hence, we can achieve high frame rates only using a simple embedded computer. Moreover, for a referencing system to be a reliable measurement for evaluating other systems, its own measurements must be verified beforehand. Thus, we have developed a methodology which employs a simple 3D-printable module to verify the system's accuracy using geometrical relationships.

II. Related work

Nowadays fiducial markers such as AprilTags [7] are commonly used [8] for mutual localization. One of the advantages of this method is that every tag can have a specific ID. However these patterns require relatively high computational resources and are susceptible to environment illumination changes and camera's rolling shutter effects. Therefore, the cameras used for these techniques must feature global shutter sensors to avoid distortion effects of rolling shutter systems which can destabilize the tracking and feature extraction systems. Nevertheless, the system introduced in this paper mitigates the rolling shutter problem by reducing the exposure time. This is feasible since the camera is highly sensitive to IR light exclusively passed through the visible light filters in the front of the camera.

In [1] multiple IR LEDs are used alongside a high speed global shutter camera in order to estimate the pose of a quadcopter using monocular vision. There are other works such as [2] where passive markers are used for detecting specific parts of a device in the global frame for performing pose estimation. However these methods are not as robust as the proposed system since a cluttered environment could potentially introduce a great amount of noise which has been eliminated in our system by the usage of visible light filters. Moreover, we achieve great results using just an inexpensive off-the-shelf rolling shutter camera and a single-board computer. High accuracy is obtained using this simple and inexpensive hardware, and furthermore, much higher frame rate is achieved. In [9] IR and RGB cameras alongside IR LEDs are used where RGB color descriptors at the vicinity of LED's centers are compared and the correspondences are derived. This is a suitable and robust method for finding the correspondences between the LEDs in the global frame and their projections on the image plane. Then again, this method is sensitive to rolling shutter effects and the processing overhead is also relatively high.

One of the most common pose estimation devices in robotics are motion capture systems such as Vicon[1]. These commercial systems use multiple cameras (fixed in the environment) and markers to estimate the pose. Even though these devices can provide outputs at high frequencies(up to 300Hz) and great precision, they are proprietary, expensive and in some cases they require numerous markers in order to provide an accurate estimation which curbs its applicability in physically constrained robotic applications.

There are numerous center-of-mass algorithms [11]–[14] for finding the center of LEDs in the image taken by the camera. These include center of gravity (COG), averaging techniques, etc. There are also methods [10] that fit circles to the dots seen in the images that would help deriving the center of the LEDs. Even though achieving a more accurate estimate of this center would allow us to have a more accurate estimation of the pose, the introduced extra processing overhead is not worth the benefits and in many cases achieving higher frame rates is preferred. On the other hand, through exploiting IR filters and thresholding techniques alongside tuning the camera's parameters (exposure time, auto white balance, etc.) desirable images of the LEDs may be retrieved in which the features are almost circular even at highly dynamic motions.

The contribution of this paper is two fold. Firstly, we have developed a simple and inexpensive referencing system that provides accurate pose measurements at a high rate. Secondly, we introduce a very simple but effective method for quantifying the accuracy and eligibility of the system requiring no special equipment. Moreover, the source codes and design files for the system are all available at our GitHub page[2] to the community.

This paper is organized as follows. In section (3) we give a brief introduction to the camera pinhole model and we review the pose estimation methods and their applications in our robot alongside investigating solutions for tracking the markers. Next in section (4), we explore the sensitivity of the algorithms to pixel quantization and measurement noise through some simulations. Subsequently, we present our method for verifying the system's accuracy eligibility in section(6) and in section (7) we comment on the concluding remarks and future work.

III. Algorithm

In this paper we use Perspective n Point (PnP) algorithm for estimating the pose of the camera. In this algorithm, at least 4 markers with known configurations are observed by a camera. After deriving the correspondences between these 3D known coordinates and their projections on the image plane, we solve for translation and orientation. This may be done using various methods such as nonlinear optimization and linear solutions through the SVD algorithm. In this paper we use 4 IR LEDs to obtain an estimate of the pose. Here, we first we explore the mathematics behind the PnP algorithm and then we investigate available methods for deriving a pose estimate for the camera.

A. Camera Pinhole Model

The camera pinhole model defines the relationship between points in the 3D world and their respective projection in the 2D world of the camera image, i.e. it is defined by the transformation matrices that would rotate and translate the points on the 3D world coordinates to the points on the camera local coordinates. The following equation represents the camera pinhole model's equation:

$$\begin{bmatrix} x \\ 1 \end{bmatrix} = K[R|t] \begin{bmatrix} X \\ 1 \end{bmatrix} \tag{1}$$

where X is mapped to the point x. K matrix represents the transformation of 3D world coordinates to homogeneous 2D

[1]http://www.vicon.com/

[2]https://github.com/aras-cdrpm-projects/IR-Referencing-System.

image coordinates which is parameterized by Hartley and Zisserman [15] as:

$$K = \begin{bmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

in which f_x and f_y represent the focal length of the camera and x_0 and y_0 are the principal point offset parameters. This point is the location of the principal point relative to the sensors origin. Furthermore s represents the axis skew. Axis skew causes shear distortion in the projected image. This distortion is caused by the imperfect camera and lens alignment or the rectangular shape of the pixels in the camera sensor.

Based on the definitions given in the homography section of this paper one can see that $K[R|t]$ is basically a homography between points in the global frame and the 2D coordinates of the same points in the image frame. Hence if we wanted to derive the orientation between the global frame and the image plane while knowing the coordinates of a few points in the global frame and image plane we need to solve (1). It should be noted that the matrix K's parameters are constants and may be derived offline.

B. The PnP Problem

The most common simplification to solving (1) is to assume known calibration parameters which is the so-called Perspective-n-Point problem as is shown below:

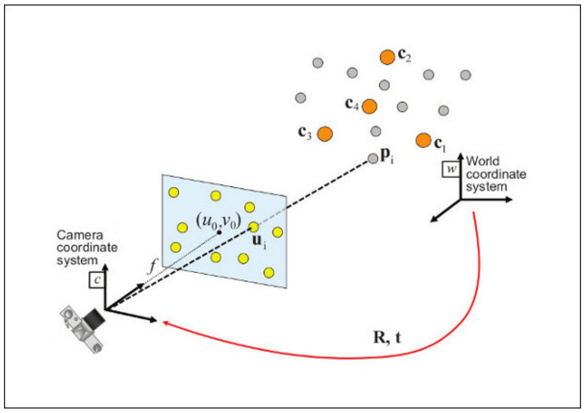

Fig. 2. Visualization of the PnP problem [18]

Let's assume we have two points in the global frame and image frame called x and X respectively. The homography between these points can be represented as follows:

$$x = PX \tag{3}$$

Assuming parameterized coordinates for x we have:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} X \tag{4}$$

As mentioned before there are numerous approaches for solving equations in the form of (4) and deriving the P matrix such as the EPnP [16] solution, singular value decomposition (SVD), optimization based approaches where optimization-based techniques are used to minimize the reprojection error. Generally, methods for estimating the pose of a camera using solutions to the PnP problem are split into two categories. One uses non-linear optimizations to get an estimate of the pose through iterations on the given data and the other focuses on the efficiency of the algorithm. In Table I we can see the characteristics of each of the used methods to derive the pose and solve the PnP problem.

TABLE I
PnP Solutions Used in this paper

Method	Iteration	Num. of Markers
EPnP [16]	Non-Iterative	≥ 4
DLS [17]	Non-Iterative	≥ 3
OpenCV PnP [18] (Proprietry)	Iterative	≥ 3
P3P [19]	Non-Iterative	$= 4$
AP3P [20]	Non-Iterative	$= 4$

In our framework we are able to switch between these methods, depending on the application in hand and suitable update rates. On the other hand, we should note that in this paper we have used the minimum number of LEDs for performing the estimation (The minimum number of LEDs required to get a proper estimation while avoiding ambiguities in the top half of the XYZ frame is 4). This is while we can introduce redundant markers and by this means, increase the accuracy while using the faster methods. By all means, we performed the experiments using all of these methods and we report the accuracy and measurement rate in Table 2 and Table 3, respectively.

C. K-Nearest Neighbors Clustering

The coordinates of markers in the global frame are always known (the configuration of the markers are known). However, we need to find the coordinates of corresponding images of these markers at every frame. To do this we use the K-Nearest neighbors clustering method (KNN) to extract positions of markers on image plane. We refer the reader to [21] for a full explanation of this method, here will only provide the pseudo code that is used in KNN in Algorithm 1.

Algorithm 1 KNN Clustering Algorithm

```
Require: Retrieve a new frame
  Do(For Every Pixel):
  1- A positive constant k is chosen as the number of clusters.
  2- The positions for k clusters are initiated from the position in the last iteration.
  3- We find the most common classification for this pixel based on the position of that pixel relative to each cluster center.
  4- We move the centers according to how the members of a cluster are spread out relative to the center of that cluster.
```

D. Marker Tracking

Now that we have identified the centers of 4 markers on the image plane, we need to label them. This means that we need to know which center corresponds to which marker in the global frame. Since there are 4 markers we will name them ”UR, UL, LR, LL” for ”Upper Right, Upper Left, Lower Right and Lower Left”.

In this paper, in order to identify the labels for each of the centers we start the camera in a known orientation. Hence, using a simple min-max method, we can label each of the centers derived from the clustering algorithm. In this algorithm first we find the center of the rectangle that is formed by the image of the markers and then find the center which has the highest x and y values with respect to the center of the rectangle and label this point as ”UR”. We do the same for the other 3 points. Even though we can use this exact method for other consecutive frames, this method will fail when the rotation of the camera in the yaw axis is more than a certain degree which leads the ”LR” marker to be detected falsely, as ”UR”.

To solve this problem we propose the following solution. At the initialization stage, we find the labels corresponding to the markers by initializing the tracking from a roughly known orientation. Then at each preceding frame, we calculate the following for every new center that is derived from the clustering algorithm.

$$\forall C_i^*, i = 1, 2, 3, 4 \quad F(C_i^*, C_i) \tag{5}$$

Where C_i^* are the new centers derived from the new frame and C_i are the coordinates of the centers that were found in the last frame which are already labeled and F calculates the Euclidean distance between every new center found in the new frame with respect to each center of the last frame. Now we have a set of 4 label-less marker positions from the current frame and 4 labeled ones from the last frame and the distance from every center of the new frame to every center from the last frame. Since the frame rate of the camera in our system is high enough, the marker positions for each one of the labels do not move by a great amount throughout consecutive frames. Hence, for every center C_i^* in the new frame we perform the steps in Algorithm 2.

Algorithm 2 Tracking algorithm

```
Require: Initialize and retrieve centers from last frame
  Do(For every new center):
  - Calculate d = $F(C_i^*, C_i)$
  if argmin(d) == $F(C_i^* - C_{UpperRight})$ then
      $C_i^*$ ← Upper Right Label
  else if argmin(d) == $F(C_i^* - C_{UpperLeft})$ then
      $C_i^*$ ← Upper Left Label
  else if argmin(d) == $F(C_i^* - C_{LowerRight})$ then
      $C_i^*$ ← Lower Right Label
  else if argmin(d) == $F(C_i^* - C_{LowerLeft})$ then
      $C_i^*$ ← Lower Left Label
  end if
```

There are various trackers implemented in image processing libraries that can be used as an alternative of this approach. However, all of these methods have computation overheads which lower the frequency of the pose estimation system. Furthermore, trackers increase complexity of the estimation algorithm. In other words, since we have simplified the frame to be processed by employing a visible light filter, there is no point in using advanced trackers and the simple algorithm we introduced in this paper accomplish the task quite robustly and accurately.

One case where the tracking algorithms shall be used to infer the labels for each marker is when ambiguities are observed. This happens when camera goes through the XY plane and looks up at the markers. Throughout this motion when looking at the edge of XY plane (when the camera is lying on the XY plane looking towards the origin, and at the markers) the markers could overlap and the proposed tracking method would fail. This situation demands a tracker that exploits a motion model for the camera and would act as a fault detection method in such situations. However, it should be noted that in stationary robotic applications such as the one implemented in this paper such cases would not occur. The marker board is placed in a position such that the camera would not pass the board itself.

IV. Simulations

In order to assess the accuracy of the proposed method and its sensitivity to noise and pixel quantization, we set up a simulation system in Matlab software. In the simulation we assume that we have 4 LEDs at the corners of a rectangular plane with dimensions of 0.1 by 0.2 meters. Then we assume a known rotation and translation matrix between the LED plane and the camera and use this to find the image of the LEDs as would be captured by the camera. These coordinates are the ground truth as seen by an ideal camera. Given this coordinates we perform the following:

- Add white noise with varying variances to the ground truth pixels on the image plane.
- Use PnP solutions and the noisy image to find the rotation and translation matrix.
- Compare the resulting transformation with the known initial ground truth. Ideally, these two sets of transformations must by identical.

When the noise added to the simulations has a variance of 2 pixels and a mean of 0 we achieve an average error of $0.3055, 0.0425$ and 0.3907 millimeters for x, y, z translations, respectively. Error values for orientation are $0.0062, 0.3522, 0.1360$ degrees for roll, pitch and yaw angles respectively. The result of this process may be seen in Fig. 3.

V. Experimental Results

We have implemented the proposed pose estimation system on a 2RT spherical parallel robot called ARAS-DIAMOND [6] shown in Fig. 1. In order to implement the algorithm, we have used a Raspberry Pi 3B mini-PC and the Raspberry Pi camera V1. The code has been written in python and OpenCV, while

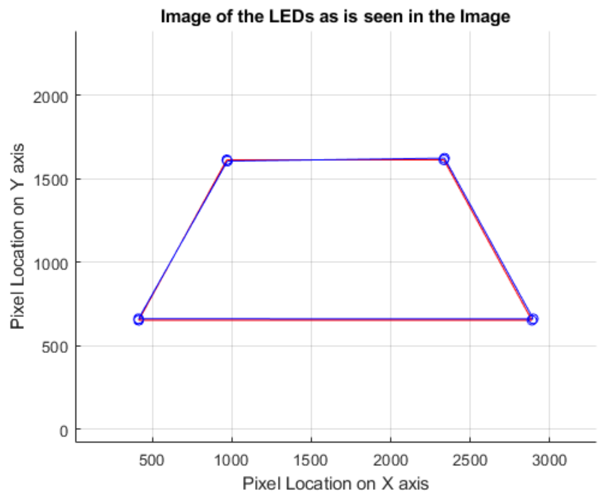

Fig. 3. The reprojection of the rectangular plane in the pixel coordinates. The blue plane represents the reprojection of the rectangular plane using the ground truth while the red plane is reprojected based on the estimated pose.

Raspicam libraries is used to retrieve images and perform the necessary calculations. Using this off-the-shelf hardware we have achieved a high frame rate of 130Hz. The source code and further details regarding the system are available on ARAS lab's GitHub page [22].

The system proposed in this paper is a referencing measurement device whose accuracy and precision must be verified before it can be used as an evaluator for other algorithms. To verify these results from our system, we employ a simple yet accurate geometrical approach using a laser pointer and a 3D printed module. This module, which is shown in Fig. 4, was mounted on a shaft with bearings to allow the module to rotate freely about the shaft. The laser pointer and the camera are installed on each side of the module and can rotate with it. As it can be seen in Fig. 5, the camera is pointed towards the LED markers, while the laser is faced towards a wall located at a far distance of 23 meters. As we rotate the module, we get two measurements, one from the camera and the other form the displacement of the laser pointer on the wall. Taking the laser system with such long distance between the laser source and the display wall as an accurate reference, we may evaluate the eligibility of the camera system for orientation estimation. As for the validation of the outputs for the translations, we tested the system on a grid of 500mm by 500mm. This grid was divided into squares of size 100mm by 100mm and 10mm by 10mm. After moving the marker around, while the camera is stationary, output from our framework was tested against the actual displacement based on the movement of camera on the grid. The setup for this system is shown in Fig. 6.

The accuracy of the proposed system is reported in Tables II and III, respectively. Based on the reported numbers in Table II, it can be seen that the iterative method from the OpenCV [18] library has the highest accuracy while requiring the longest time to perform necessary calculations. This is while other methods such as P3P [19] and AP3P [20] are faster with respect to computational time, while yielding lower accuracies. To achieve higher accuracies using the EPnP [16] and DLS [17] methods, one would require a larger number of markers alongside a RANSAC [23] algorithm to reject outliers and hence they are not appropriate methods for our application. As for the translation accuracies, the results do not differ by a great margin for the values acquired from various methods and the only decision factor would be the iteration frequency.

TABLE II
ORIENTATION ESTIMATION RESULTS

$Method$	Average Absolute Error (deg)	Average Est. Time (ms)
EPnP	1.80342	0.936
DLS	1.91864	0.955
OpenCV Iterative	**0.06926**	**2.4246**
P3P	0.418	0.3130
AP3P	0.418	0.2779

TABLE III
TRANSLATION ESTIMATION RESULTS

$Method$	Average Absolute Error (mm)	Average Est. Time (ms)
EPnP	1.17	0.936
DLS	1.03	0.955
OpenCV Iterative	0.954	2.4246
P3P	1.11	0.3130
AP3P	0.97	0.2779

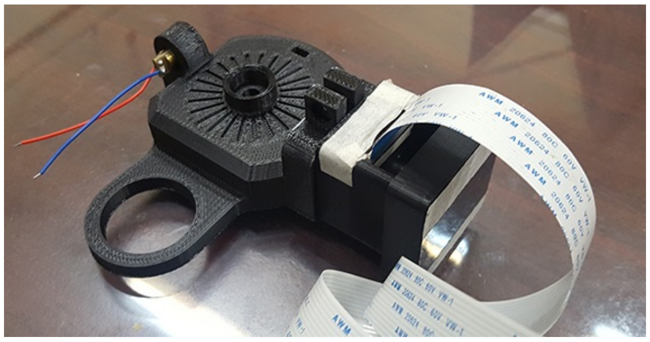

Fig. 4. The referencing module.

VI. CONCLUSION

In this work, we have developed a referencing solution for general stationary and mobile robots using low-cost hardware and 3D printable components. Even though the system is simple and inexpensive, it provides measurements that are comparable to commercial and expensive systems like Vicon. Furthermore, The proposed method is robust against cluttered environments and noises from surroundings and works at a frequency of 130Hz. This solution uses a single camera and a number of IR LEDs to accurately determine the rotation and translation of the system with respect to a global frame. We have made the source codes and design files of our system

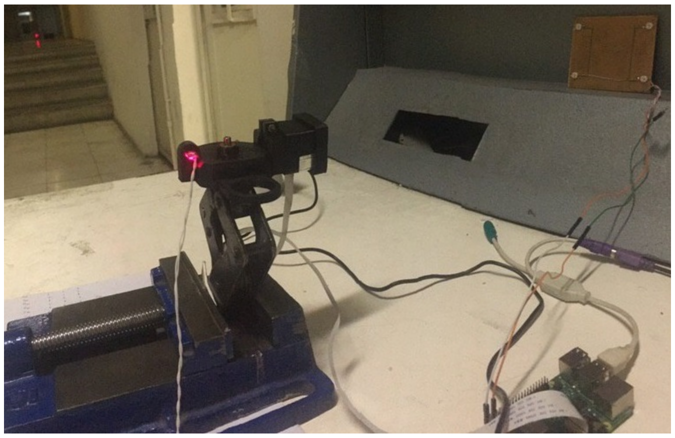

Fig. 5. The setup for verification procedure.

publicly available for the community on our GitHub page[3]. Based on the obtained results it is evident that there is a trade-off between the achieved accuracy and the frequency at which the pose can be estimated in an online application. Basically, more computationally expensive methods require the longest time to perform necessary computations while reporting the highest accuracy. Among the numerous methods for pose estimation, we have chosen the OpenCV Iterative [18] method to derive the pose of the end-effector in our application. Even though this method is the most expensive one among the available implementations in the OpenCV library, it is able to provide estimates at rates higher than 100Hz which is suitable for our application.

In the future, we plan to compensate for the intrinsic delay in the image pipeline using predictive filters and additional sensors such as IMUs. Furthermore we plan to add additional cameras to the system in a modular way to increase the working area of the system and improve its accuracy by combining the measurements in a statistical framework.

References

[1] M. Faessler, E. Mueggler, K. Schwabe and D. Scaramuzza, "A monocular pose estimation system based on infrared LEDs," *2014 IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, 2014, pp. 907-913. doi: 10.1109/ICRA.2014.6906962

[2] A. Breitenmoser, L. Kneip and R. Siegwart, "A monocular vision-based system for 6D relative robot localization," *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA, 2011, pp. 79-85. doi: 10.1109/IROS.2011.6094851

[3] E. Walsh, W. Daems and J. Steckel, "An optical head-pose tracking sensor for pointing devices using IR-LED based markers and a low-cost camera," *2015 IEEE SENSORS*, Busan, 2015, pp. 1-4. doi: 10.1109/ICSENS.2015.7370112

[4] M. Maidi, JY. Didier, F. Ababsa, et al, "A performance study for camera pose estimation using visual marker based tracking," *2010 Machine Vision and Applications*, 2010, pp. 365-376. doi: 10.1007/s00138-008-0170-y.

[5] E. Marchand, H. Uchiyama and F. Spindler, "Pose Estimation for Augmented Reality: A Hands-On Survey,". *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 12, pp. 2633-2651, 1 Dec. 2016. doi: 10.1109/TVCG.2015.2513408

[6] A. Molaei, E. Abedloo, H. D. Taghirad and Z. Marvi, "Kinematic and workspace analysis of DIAMOND: An innovative eye surgery robot," *2015 23rd Iranian Conference on Electrical Engineering*, Tehran, 2015, pp. 882-887. doi: 10.1109/IranianCEE.2015.7146336

[7] E. Olson, "AprilTag: A robust and flexible visual fiducial system,". *IEEE International Conference on Robotics and Automation*, Shanghai, 2011, pp. 3400-3407. doi: 10.1109/ICRA.2011.5979561

[8] G. Zhenglong, F. Qiang and Q. Quan, "Pose Estimation for Multicopters Based on Monocular Vision and AprilTag," *Proceedings of the 37th Chinese Control Conference (CCC)*, 2018, Wuhan, China, pp. 4717-4722. doi: 10.23919/ChiCC.2018.8483685

[9] G. Koutaki, S. Hirata, H. Sato and K. Uchimura, "[POSTER] Marker Identification Using IR LEDs and RGB Color Descriptors," *2015 IEEE International Symposium on Mixed and Augmented Reality*, Fukuoka, 2015, pp. 96-99. doi: 10.1109/ISMAR.2015.30

[10] D. Umbach and K. N. Jones, "A few methods for fitting circles to data," *IEEE Transactions on Instrumentation and Measurement*, vol. 52, no. 6, pp. 1881-1885, Dec. 2003. doi: 10.1109/TIM.2003.820472

[11] L. Zhang, G. Chen, D. Ye and R. Che, "A Fast Center of Mass Estimation Algorithm for Coordinates of IR Markers," *2008 The 9th International Conference for Young Computer Scientists*, Hunan, 2008, pp. 1120-1125. doi: 10.1109/ICYCS.2008.506

[12] N. M. Yu, T. Shibata and T. Ohmi, "A real-time center-of-mass tracker circuit implemented by neuron MOS technology," *in IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 45, no. 4, pp. 495-503, April 1998. doi: 10.1109/82.663806

[13] P. Chen and W. W. Seemuller, "Center of Mass Measurement Using an Array System," *in IEEE Transactions on Instrumentation and Measurement*, vol. 22, no. 1, pp. 78-83, March 1973. doi: 10.1109/TIM.1973.4314102

[14] A. Fish, D. Akselrod and O. Yadid-Pecht, "An adaptive center of mass detection system employing a 2-D dynamic element matching algorithm for object tracking," *Proceedings of the 2003 International Symposium on Circuits and Systems*, 2003. ISCAS '03., Bangkok, 2003, pp. III-III. doi: 10.1109/ISCAS.2003.1205135

[15] R. Hartley and A. Zisserman, "Multiple view geometry in computer vision," 2nd edition, *Cambridge University Press*, March 2004, ISBN: 0521540518.

[16] V. Lepeti, F. Moreno-Noguer and P. Fua, "EPnP: An Accurate O(n) Solution to the PnP Problem," *International Journal Of Computer Vision. 2009*, February 2009, pp. 81:155. doi: 10.1007/s11263-008-0152-6

[17] J. A. Hesch and S. I. Roumeliotis, "A Direct Least-Squares (DLS) method for PnP," *2011 International Conference on Computer Vision*, Barcelona, 2011, pp. 383-390. doi: 10.1109/ICCV.2011.6126266

[18] OpenCV Documentation, *Camera Calibration and 3D Reconstruction*, Accessed on: 20 Jul. 2019 [online] Available: https://docs.opencv.org

[19] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang and Hang-Fei Cheng, "Complete solution classification for the perspective-three-point problem,". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 930-943, Aug. 2003. doi: 10.1109/TPAMI.2003.1217599

[20] T. Ke and S. I. Roumeliotis, "An Efficient Algebraic Solution to the Perspective-Three-Point Problem," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 4618-4626. doi: 10.1109/CVPR.2017.491

[21] V. Garcia, E. Debreuve and M. Barlaud, "Fast k nearest neighbor search using GPU," *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Anchorage, AK, 2008, pp. 1-6. doi: 10.1109/CVPRW.2008.4563100

[22] ARAS-IREF GitHub, *ARAS IR Referencing System*, Accessed on: 30 Jul. 2019 [online] Available: https://github.com/aras-cdrpm-projects/IR-Referencing-System.

[23] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6 , pp. 381-395. doi: 10.1145/358669.358692

[3]https://github.com/aras-cdrpm-projects/IR-Referencing-System