Enhanced Precision in Built Environment Measurement: Integrating AprilTags Detection with Machine Learning

Shengtao Tan[1], Aravind Srinivasaragavan[1], Kepa Iturralde[1,2], Christoph Holst[1]

[1]Chair of Engineering Geodesy, School of Engineering and Design,
Technical University of Munich, 80333 Munich, Germany
[2]Chair of Digital Transformation in Construction, Institute of Construction Management,
Faculty of Civil and Environmental Engineering, University of Stuttgart, 70569 Stuttgart, Germany.

shengtao.tan@tum.de, aravind.srinivasaragavan@tum.de kepa.iturralde@tum.de christoph.holst@tum.de

Abstract -

In the field of building renovation with prefabricated modules, accurately locating and identifying connectors' positions and orientations is an essential technological challenge. For building renovation with prefabricated modules, traditional methods like total stations are not only time-consuming but also highly dependent on experienced technicians. However, previous research has proven that ApriTtag tags can be effectively used in building measurements. This paper proposes a refined AprilTag detection pipeline that integrates machine learning techniques, significantly improving detection accuracy. Moreover, this process can be easily used by non-experts making it more accessible and less time-consuming.

Keywords -

AprilTag; Machine Learning; Neural Network; Building Measurement

1 Introduction

Existing building stock renovation is a critical aspect of construction automation. For instance, accurately determining the position of connectors is essential for the installation of prefabricated panels on building exteriors. However, traditional measurement methods, such as the use of total stations(see [1]), have several limitations, including being time-consuming, requiring skilled technicians, etc. To address these challenges and leverage the advantages of automation, the integration of computer vision techniques with a visual fiducial system is suggested as a viable solution. In this paper, we propose a refined Apriltag(see [2]) detection pipeline integrated with machine learning to tackle the above problem. Figure 1 shows its architecture. We will first discuss the research gaps in the accuracy of AprilTag localization. Then we will introduce the components of the pipeline in detail. The experiments in section 4 show that our refined pipeline has very good accuracy. This research is part of the ENSNARE project[3].

2 Research gaps

The AprilTag is widely applied across various domains, including robot navigation & localization, industry automation, and augmented reality. However, existing research puts limited focus on localization accuracy when employing AprilTags. López-Cerón et al.[4] conducted an analysis of AprilTag's accuracy, but their study was limited to camera-to-target distances of no more than 6 meters. Kallwies et al.[5] extended their testing range to 18 meters, yet their focus was on pixel-level errors, rather than millimeters. Similarly, Olson et al.[6] investigated distances up to 80 meters, but their error tolerance is in meters. Additionally, research on the large-scale layout of AprilTags is very scarce. Kallwies et al.[5] constructed a simulated 7 x 22 grid with 152 AprilTags, each 13 cm in size. Beyond this study, there is a lack of research exploring the potential and challenges of large-scale AprilTag layouts. To sum up, there is a notable gap in research and applications concerning the combination of high accuracy and the large-scale layout of AprilTags. But in our context, we mostly applied AprilTags in large-scale outdoor environments and want to achieve accuracy that is comparable to or even beyond traditional methods like total-station measurement. Therefore, this paper aims to bridge this gap.

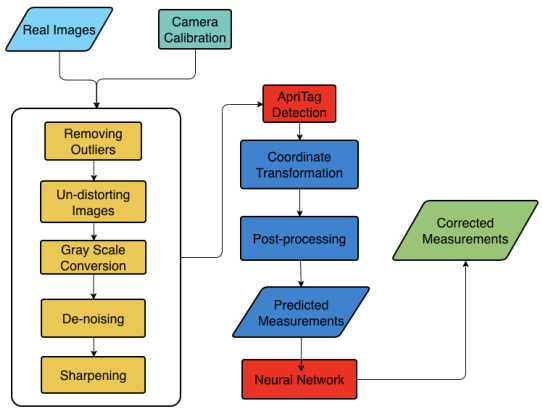


Figure 1. A flowchart outlining the pipeline

3 AprilTag detection pipeline

As in previous experiences with OpenCV and AprilTags, we **calibrated a camera**, in our case a Sony A7R4 [7] with different lenses using a checkerboard in ambient lighting conditions. For each focal length, we took pictures in different camera positions and orientations while making sure that the checkerboard eventually covered the entire image frame. Finally, we loaded all the pictures to the MATLAB R2023b Camera Calibration toolbox and only left around 15 pictures that had the smallest pixel errors as our final calibration candidates.

Before using the AptilTag detector, we first **Preprocess the images** with the following steps:

1. **Removing outliers:** We filter out the pictures that have undetected tags (false negative) and non-existent tags (false positive).
2. **Minimizing image distortion:** cv2.undistort function is applied to undistort images using the camera calibration and distortion matrix from the calibration step. This step can increase the detection rate.
3. **Grayscale conversion:** We convert the images to the acceptable format of AprilTag Detector, which is grayscale.
4. **De-noising:** A Gaussian filter with kernel size 5 x 5 is applied to reduce the sensor noise of the camera.
5. **Sharpening:** We subtract the smoothed image of the last step from the original image

For the **AprilTag detection** algorithm, we used the AprilTag3(see [8]) library of Python. The motivation is that the Python bindings allow efficient development with powerful libraries, such as NumPy. We mostly use the default parameters of the AprilTag3 binder.

Using a **Coordinate transformation** one can express the pose of tags in the coordinate frame built up at the center of the bottom left tag. To do so, we first obtain the rotation matrix and translation vector for each tag from the tag frame to the camera frame using cv2.SOLVEPNP_IPPE_SQUARE function. With the bottom left tag as tag 1 and the other as tag 2, tag 2's position and orientation(in the new coordinate system) can be expressed using the formulas below:

$$P_{tag2} = R_1^T * (t_2 - t_1) \tag{1}$$

$$R_{2|1} = R_1^T * R_2 \tag{2}$$

where R_1 is the rotation matrix of tag 1. t_2 and t_1 are the translation vector of tag 2 and tag 1 respectively. $R_{2|1}$ is the rotation matrix of tag 2 w.r.t tag 1. The Euler angles of tag 2 can be further computed from $R_{2|1}$.

A **Postprocessing** was necessary because, we took 10 to 200 pictures of the tags, which means we have lots of measurements. In the end, we only need one final and accurate measurement. Therefore, the postprocessing is a crucial step of the whole pipeline. We took two strategies for postprocessing:

1. **Mean values:** The mean value is used when there are not many measurements or DBSCAN can't find a valid dense area of the measured point cloud.
2. **DBSCAN clustering:** By visualizing the positions of detected tags' center, there are usually some outliers and also a dense area of point cloud (See Figure 2). To filter out the outliers, we use DBSCAN(see [9]) to find the core measurements.

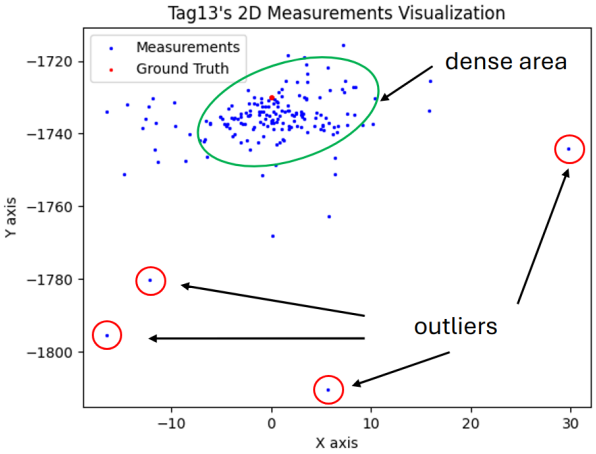


Figure 2. Point cloud visualization

We also implemented two other strategies based on the ranking of pose_err and blur scores.

1. **Pose_err ranking:** pose_err represents the object-space error of the estimation. The idea is to only select the tag measurements that have lower pose_err.
2. **Blur score ranking:** Blurry images can contribute to poor detection accuracy. To avoid blurry images, we first mask the pictures and only keep AprilTags because those squares are our ROIs(Region of Interest). Then we compute the total variance of the laplacian of ROIs using cv2.Laplacian as the blur score for the image. Eventually, we only compute the mean of less blurry pictures.

The performance comparison of the above 4 strategies will be introduced later.

3.1 Machine learning correction

The AprilTag detection pipeline is highly susceptible to lighting conditions which can influence its performance and these "noises" are hard to determine and control. We have devised a neural network trained on a dataset of hundreds of images, such as the one shown in Figure 4c,

taken in different lighting conditions and positions. Essentially, it is trained to predict the correct ground truth values based on the experimental observations. The idea is that the neural network can then improve on the values predicted by the AprilTag detection pipeline. The dataset consists of ground truth values of different AprilTags in different images such as their 3d coordinates as well as camera information. Our approach is inspired by [10].

The neural network is implemented in Python version 3.10.10 using Tensorflow(See [11]). We have used a standard fully connected feed-forward network with 2 hidden layers as shown in Figure 3.

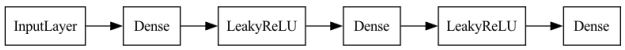
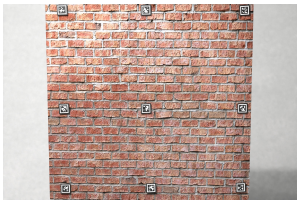


Figure 3. Network Architecture

The dataset is first randomly shuffled to increase the uniformity of the dataset. We split the dataset randomly into a training set and a testing set. The size of the training set is 95% of the original dataset. We then initialize the neural network with the following parameters: we use Adamax for the optimization process along with l2 regularization with regularisation constant 0.01234. We use mean absolute error for the loss function and finally, we set the batch size to 40 and train the model for 200 epochs.

4 Experiments and results

The proposed pipeline was tested in simulation, indoor and outdoor environments respectively.



(a) simulation

(b) indoor

(c) outdoor

Figure 4. Example pictures of real-world and simulation experiments

4.1 Simulation testing

For simulation, we use Blender(See [12]) to generate synthetic images (See Figure 4a). The first part of Table 1 shows that using mean values or DBSCAN outperforms the rest two postprocessing methods. In addition, without the influence of environmental noise, our pipeline reached millimeter-level accuracy.

Table 1. The results of proposed pipeline

Simulation result				
	Mean	DBSCAN	Pose_err	Blur score
Δx/mm	**0.40**	0.40	0.40	0.40
Δy/mm	-0.45	**-0.36**	-0.45	-0.45
Δz/mm	**-0.10**	0.10	-0.10	-0.10
Δxy/mm	0.60	**0.54**	0.60	0.60
Δxyz/mm	0.61	**0.55**	0.61	0.61
$\Delta\alpha$/ °	**0.04**	0.07	0.04	0.04
$\Delta\beta$/ °	**0.03**	0.05	0.03	0.03
$\Delta\gamma$/ °	**0.00**	0.00	0.00	0.00
Indoor experiment				
	Mean	DBSCAN	Pose_err	Blur score	Total station
Δx/mm	0.82	**-0.43**	1.50	1.71	0.60
Δy/mm	2.23	**1.31**	2.97	1.49	-0.20
Δz/mm	-5.59	-5.37	**0.01**	-8.29	-1.29
Δxy/mm	2.38	**1.38**	3.33	2.27	0.63
Δxyz/mm	6.07	5.54	**3.33**	8.59	1.44
$\Delta\alpha$/ °	-0.71	-0.38	**-0.30**	-1.07	-
$\Delta\beta$/ °	**-0.84**	-1.07	-1.03	-1.14	-
$\Delta\gamma$/ °	**0.01**	0.02	0.04	0.03	-
Outdoor experiment					
	Mean	DBSCAN	Pose_err	Blur score	Total station
Δx/mm	4.42	**1.87**	3.24	7.22	1.15
Δy/mm	-4.27	**-4.20**	-2.60	-11.72	1.00
Δz/mm	19.83	**11.84**	16.74	22.11	-1.29
Δxy/mm	6.15	4.60	**4.15**	13.77	1.52
Δxyz/mm	20.76	**12.70**	17.25	26.04	2.00
$\Delta\alpha$/ °	0.73	1.65	**0.07**	1.43	-
$\Delta\beta$/ °	1.08	**0.49**	0.82	2.96	-
$\Delta\gamma$/ °	0.10	**-0.06**	0.10	0.20	-
Neural network correction					
	Indoor coorection		Outdoor correction	
	Mean	DBSCAN	Mean	DBSCAN
Δx/mm	0.76	-0.42	4.00	-0.25
Δy/mm	-3.08	-2.37	-7.33	-8.22
Δz/mm	2.56	1.94	5.50	4.44
Δxy/mm	3.17	2.41	8.35	8.22
Δxyz/mm	4.08	3.09	10.00	9.35
$\Delta\alpha$/ °	0.37	0.48	1.52	-0.29
$\Delta\beta$/ °	-2.85	-2.35	-5.24	-1.93
$\Delta\gamma$/ °	3.73	3.05	5.38	2.96

4.2 Real-world testing

Building a large-scale testing environment is not only challenging, but it is also impractical to test the entire building's exteriors with exact ground truth. To mitigate this, we used a 2000 x 2000 millimeters calibrated wooden board as a substitution for placing our AprilTags. Each tag is 15 cm in length and is augmented with four reflective tapes at its corners, for measurements with a total station. The AprilTags were accurately placed with rulers.

Regarding the total station measurements, we used the Leica TC702. For better accuracy of the total station results, each layout was measured from 2 to 3 different positions in two phases.

The **indoor experiments** were carried out in a lab with shooting distances ranging from 4 to 15 meters. Figure 4b shows an example of the indoor picture.

The second part of Table 1 shows the accuracy of a representative indoor example layout. Bold numbers are the best results among 4 postprocessing methods.DBSCAN still performs the best in general, but pose_err also shows good results. Our proposed pipeline reaches the total-station-level accuracy.

The **outdoor experiments** are carried out on the campus with shooting distances ranging from 5 to 18 meters. The third part of Table 1 shows the accuracy of a representative outdoor example layout. One major reason that decreased the accuracy could be the poor illumination of the experiment environment. Among all the postprocessing methods, DBSCAN also achieved the best performance.

The **neural network** was tested on the test dataset and performed quite well with a mean absolute loss of 3.5547mm and an accuracy of 0.7551. The last part of the Table 1 shows a sample correction result. We can see there's an improvement in the depth measurement.

Further, we tracked the consumed time for our pipeline in the experiments. Table 2 shows that our method is much more efficient than the total station while achieving similar accuracy.

Table 2. Spent time comparison

	AprilTag detection	total station
setting up/min	**3**	13
measuring/min	**15**	28

5 Conclusion

The research described in this paper has successfully demonstrated a collection of techniques that allow for an accurate and faster process of building measurements, especially compared to conventional methods. Future research will include:

1. Improving the machine learning algorithm. We are currently exploring the possibility of using 3D rendering software to generate synthetic images through which we can generate more datasets.

2. Integrate and test the proposed pipeline with UAVs. We will work closely with another team on this project that is currently developing a UAV for sticking the AprilTags.

In the next steps of the research, the idea is to use this technique in real buildings.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 958445.

References

[1] R. Volk, J. Stengel, and F. Schultmann. Building information modeling (bim) for existing buildings — literature review and future needs [autom. constr. 38 (march 2014) 109–127]. *Automation in Construction*, 38:109–127, 2014. doi:10.1016/j.autcon.2013.10.023.

[2] K. Iturralde, J. Shen, and T. Bock. Apriltag detection for building measurement. In *Proceedings of the 40th ISARC*, pages 589–592, Chennai, India, 2023. IAARC. ISBN 978-0-6458322-0-4. doi:10.22260/ISARC2023/0079.

[3] ENSNARE. `https://www.ensnare.eu/`, Accessed: Dec. 13, 2023.

[4] A. López-Cerón and J. Cañas. Accuracy analysis of marker-based 3d visual localization. pages 1124–1131, 2022. doi:10.17979/spudc.9788497498081.1124.

[5] J. Kallwies, B. Forkel, and H. Wuensche. Determining and improving the localization accuracy of apriltag detection. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8288–8294, 2020. doi:10.1109/ICRA40945.2020.9197427.

[6] E. Olson. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3400–3407. IEEE, 2011.

[7] Sony Camera. `https://www.sony.de/electronics/wechselobjektivkameras/ilce-7rm4a/specifications`, Accessed: Dec. 19, 2023.

[8] AprilTag Detector. `https://github.com/pupil-labs/apriltags`, Accessed: Dec. 13, 2023.

[9] scikit. `https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html`, Accessed: Dec. 13, 2023.

[10] B. Mariusz, D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. LaJackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba. End to end learning for self-driving cars, 2016.

[11] Tensorflow. `https://www.tensorflow.org/`, Accessed: Dec. 16, 2023.

[12] Blender. `https://www.blender.org/`, Accessed: Dec. 13, 2023.