Image-tag-based indoor localization using end-to-end learning

Mohammed Alarfaj1, Zhenqiang Su2, Raymond Liu3, Abdulaziz Al-Humam4 and Huaping Liu2

Abstract
Image or feature matching-based indoor localization still faces many technical challenges. Image-tag-based schemes using pose estimation are accurate and robust, but they still cannot be deployed widely because their performance degrades significantly when the tag-camera distance is large, which requires densely distributed tags, and the designed system generally is specific to some special tags and lenses. Also, the lens distortion degrades the performance appreciably and is difficult to correct, especially for the wide-angle lenses. This article develops an image-tag-based indoor localization system using end-to-end learning to overcome these issues. It is a deep learning–based system that can learn the mapping from the original tag image to the final 2D location directly from training examples through self-learned features. It achieves consistent performance even when the tag-camera distance is large or when the image has a low resolution. The mapping learned by the deep learning model factors in all kinds of distortions without requiring any distortion estimation. The tag design is based on shape features to make it robust to lighting changes. The system can be easily adapted to new lenses/cameras and/or new tags. Thus, it facilitates easy and rapid deployment without requiring knowledge from domain experts. A drawback of the general deep learning model is its high computational requirements. We discuss practical solutions to enable real-time applications of the proposed scheme even when it is running on a mobile or embedded device. The performance of the proposed scheme is evaluated via a set of experiments in a real setting and has achieved less than 20 cm of positioning errors.

Keywords
Image tag, indoor localization, deep learning

Introduction
Image-based indoor localization1–10 finds many use scenarios such as drone and robot navigation. In such methods, images of the environment where localization takes place are pre-captured, structured, and stored in a database, using methods such as image matching and bundle adjustment algorithms. Each image’s relative pose such as its position and orientation is estimated and the semantic locations of the images are tagged. The camera’s position (or the position of a user) is localized by matching the camera’s captured image to one of the images in the database, for example, by combining quick image searching, feature matching (such as

1Department of Electrical Engineering, College of Engineering, King Faisal University, Al-Ahsa, Saudi Arabia
2School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR, USA
3Department of Computer Science, Princeton University, Princeton, NJ, USA
4Department of Computer Science, King Faisal University, Al-Ahsa, Saudi Arabia

Corresponding author:
Huaping Liu, School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331, USA.
Email: huaping.liu@oregonstate.edu

Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).
Scale Invariant Feature Transform (SIFT) features), and relative orientation. Image-based methods still suffer from issues such as high computational requirements, blind spot issues, and being not robust.

Image-tag-based (or marker-based) indoor localization avoids these issues using pre-defined image tags. In such methods, pre-designed tags are attached at certain known locations in the environment. If a photo taken by a camera contains any of the tags, the tag can be detected and used to determine the pose of the camera with reference to the location and pose of the tag through pose estimation (PE). This requires the camera to be calibrated properly and with a known calibration matrix. Image-tag-based methods can achieve a high accuracy when the tag is clearly present in the image. They can also provide robust and continuous localization when the tags are densely distributed in the environment.

However, as discussed in Liao and Tsai and Anwar et al., the accuracy of PE relies on the detection accuracy of the local corners or points in the image. When the image is blurred or taken from far away, or when the tag is partially covered, PE accuracy decreases. Also, camera lenses have different kinds of distortions (e.g., barrel distortion), which will degrade the localization accuracy, especially for cameras with wide-angle lens. Existing PE algorithms either do not consider such distortions, resulting in dramatically decreased accuracy as the range increases, especially for cameras with wide-angle lens, or they have considered such distortions at the expense of much increased complexity and computational needs in order to correct these distortions on the original full-size image. Also, the distortions often come from a combination of different sources and are complex and highly non-linear. The corrections would at best be an approximation and the errors are unevenly distributed over the image. These errors propagate into the next steps of the PE algorithms. While using cameras with wide-angle lenses is attractive for reducing the tag density, the resulting distortion–correction errors are more substantial. With PE algorithms, if the lenses have changed and/or if the tags have been redesigned, the algorithms must be revised accordingly, preventing them from being deployed conveniently. Another problem is that existing tags are color based for easy detection and processing, but they are not robust to lighting changes.

The results in Liao and Tsai have shown that image-tag-based localization using machine learning (ML) achieves better accuracy than using PE algorithms, especially in the long-range scenarios. This article designs an image-tag-based indoor localization using end-to-end learning, in which tags designed are placed on the walls with known IDs and positions. The camera that acquires images for positioning could be from any handheld devices or from a moving vehicle.

The proposed system resolves many of the aforementioned drawbacks and has the following unique features:

- It is a deep learning (DL)–based method with better learning abilities. The ML scheme only accepts vectors as inputs and generally relies on manually designed features. Using manually designed features requires domain knowledge and the designed features are not necessary optimized for learning the mapping from input to the label. The proposed approach uses DL to automatically learn the necessary features directly from the input images. Also, since DL models can learn any arbitrary mappings, they achieve consistent performance in cases such as when the input images are blurred, have low resolution, or are partially covered, as long as they see similar samples in the training sets.
- It takes into account all potential distortions, including the wide-angle lens distortion, via end-to-end learning. Instead of trying to estimate the distortions with few parameters as done in Wu et al., DL models are able to learn the highly non-linear mappings directly from the original tag images to the final two-dimensional (2D) position of the camera through training. Within these mappings, all distortions introduced by the lens and/or camera are represented. Therefore, the proposed scheme does not require distortion estimation, nor suffer from estimation error issues.
- The tag designed in this proposed system is shape-based, rather than color-based. Therefore, it is more robust to lighting changes in the environment.
- The proposed system is designed to be easily adapted to new lenses/cameras and/or new tags, allowing fast and easy deployment without requiring knowledge from domain experts.
- Finally, since the DL model has a higher computational need than ML, new algorithms are developed to make it suitable for real-time applications running on mobile or embedded devices.

In summary, the proposed method can learn the mapping from original tag images to the final position of camera through automatic feature extraction. And it accounts for all potential distortions when doing the mapping learning. In addition, the tags were specially designed to against the lighting change. We will discuss the proposed scheme in detail in section “Image-tag-based localization with DL,” including tag requirements and appropriate tag designs, tag segmentation and classification, distance and angle extraction, and camera position estimation using distance and angle information. Section “Experiments and results” presents a
comprehensive set of experiments in a real environment and discusses a complexity issue of the proposed system, as well as potential solutions to help achieve real-time performance even on a mobile or embedded device. A discussion on the possibility of a standard flow for the proposed system to be quickly deployed and adaptively accommodate new changes to the tags and/or lenses/cameras is also provided in this section.

**Image-tag-based localization with DL**

Figure 1 shows the process of the proposed scheme. First, $N$ different tags are designed, where $N$ should be sufficiently large for large coverage areas. Then, three convolutional neural networks (CNNs) are trained to predict the tag ID, distance, and angle, respectively. Note that these processes only need to be performed once, that is, once the tags are designed and the models are trained, they can be used anywhere. After training, the tags are distributed in the localization environment (e.g. attached on walls), and the system is ready.

For localization as illustrated in Figure 2, first, a camera automatically scans its environment. If the scanned photo contains at least one tag, then the tag(s) is segmented out and is used to estimate the tag ID, distance, and angle information via the pre-trained CNNs. Distance here refers to the distance between the camera and the tag, while angle refers to the angle between the...
line perpendicular to the tag and the line between the user and the tag, \(d\) and \(\theta\), respectively, as illustrated in Figure 2. Finally, the location of the camera is estimated using the method shown in Figure 3, where the location is an intersection of a semi-circle and a line. One tag captured is sufficient for localization; more tags captured will lead to improved accuracy.

The CNN model could map the tag image to a final 2D position of the camera directly. In this article, we estimate the distance as well as angle because the distance estimation is much more reliable than the angle estimation. Therefore, if multiple tags are present, it is better to use distances from the tags to estimate the location, just like the radio-frequency (RF) signal localization schemes do.\(^{21-30}\)

**Tag design**

To effectively extract useful information from an image of the tag through a CNN, the design of the tag should meet the following requirements: (1) it is easy to create a large number of different tags in order to cover a large area; (2) it has simple features, so that it can be seen clearly from far distances; (3) it has non-symmetric features, to distinguish positive and negative angles; (4) for reliable detection or segmentation, the features should be shape-based, not color-based. Although it is easier to segment the tag from the photo with color-based features,\(^{11-19}\) features from color are not robust since lighting variations could fail the segmentation; (5) all tags should have a common feature for tag segmentation as well as distance and angle information extraction.

There are existing tags designed for different purposes. Examples of existing tags are shown in Figure 4 and a summary of such tags is listed as follows:\(^{11-19,31,32}\)

- **QR-Code:** it uses a quadrilateral visual tag where information is stored in a binary pixel matrix of 1817–7089 characters, depending on the alphabet used. A main problem of its visual tag is its rather low detection speed, due to the large amount of information stored in it.\(^{31,33}\)
- **Maxicode:** this system, which is used in the US postal services, is a quadrilateral visual tag, where information is stored in a hexagonal binary pixel matrix. Unlike the QR-Code, its visual tag can store up to 93 characters, and thus can be detected quickly.\(^{31,34}\)
- **CyberCode:** in this system, information is stored in a binary pixel matrix. Unlike QR-Code and Maxicode, its visual tag is designed for quick and robust detection.\(^{31,35}\)
- **ReacTIVision:** this system, which is used in “Reactable,” uses a very simplistic quadrilateral visual tag, wherein the information is stored according to the shape of its internal pattern.\(^{31,34}\)
- **ARToolKit:** it is probably the first one for positionining applications. It uses a quadrilateral visual tag where information is stored according to the shape of its internal pattern. Because detection is based on recognizing the pattern of its visual tag, this pattern should be unique to be easily detectable in a natural environment, and be simple to be quickly detectable.\(^{31,34}\)
- **ARTag:** this system, a successor of the ARToolKit, uses a quadrilateral visual tag, wherein up to 36 characters of information can be stored in a binary pixel matrix, which is protected by an FEC code. Its visual tag is quick to detect since the matrix contains very little information. It is robust against rotation and perspective views.\(^{31,34}\)

The QR-Code and Maxicode are used only for near distance applications whereas other tags could be read from long distances, but they either miss a common feature or not have a non-symmetric pattern. None of these tags meet all the requirements for the proposed localization system.

An example of the tag that meets all the requirements is shown in Figure 5. The top-left, non-symmetric square that consists of small circles is designed for segmentation as well as distance and angle information extraction. Since the squares form the boundary of the tag, segmentation is done by identifying the boundary.
The reason for the use of small circles, rather than solid lines, to form the square is that the small circles enrich the features. The rich features can help distinguish the tag from the background and, therefore, simplify segmentation. However, overly complex features will cause issues; for example, they cannot be seen clearly from a far distance. The square is the common feature of all tags, but its shape and size will change when seen from different distances and angles. So, the distance and angle information is encoded in the “distortion” of the square seen in the image.

The bottom left of Figure 5 shows some features that are used to fill the inner blank space of the square to generate different tags through combination and permutation of their features. Note that many more such features can be designed. Therefore, a large number of different tags can be designed easily. The right-hand side of Figure 5 shows an example tag with filled features.

**Tag detection**

As shown in Figure 1, tag identification, and distance and angle estimations all depend on tag detection. Object detection has been a well-studied problem in DL, like the Single Shot MultiBox Detector (SSD) and You Only Look Once (YOLO) architectures, which have achieved real-time performance with high accuracy and have already been widely used in many applications. However, training these models requires extensive manually labeled examples, making them difficult for fast deployment. In addition, if the designed localization system needs to adapt to new lenses or newly designed tags, then all training samples must be relabeled manually, which is a time-consuming and labor-intensive task. To this end, instead of using SSD or YOLO models to detect the tag in the image, the template matching style method is used.

For the proposed image-tag-based localization, only the square feature is needed, rather than the exact boundary of the object. Note that the outer square feature could become a parallel, echelon, or other shapes when viewed from different angles and distances. Thus, making full use of the common feature for all tags is critical. The square as shown in Figure 5 aims to provide simple, accurate, and robust detection. It is also possible that one image could capture more than one tag. In such a case, all tags captured should be correctly detected.

A good detection should be precise and complete, as illustrated in Figure 6. The best-known template matching methods are SIFT and speeded-up robust features (SURF). An issue with SIFT is that it is not robust to changing light conditions and when the image is blurred; in such cases, it may result in incomplete and occluded segmentations. We develop an accurate and robust segmentation method that involves three steps: key feature searching, clustering, and prediction.

**Key feature searching.** Key feature searching learns the key features in the template and extracts similar features in the target image and locates them. The top plot of Figure 7 shows examples of some key features extracted (the dots). Note that part of the extracted features are caused by the environment, albeit they resemble the key features.

**Hierarchical clustering.** The second step employs the hierarchical clustering method to cluster the key features. With some proper threshold, several main clusters can be formed. The example in the middle of Figure 7 shows that there are three main clusters, labeled as c1, c2, and c3. Based on each cluster, we segment the patches out of the image with a square window that covers all key points. The segmented patches could be complete tags, incomplete tags, occluded tags, or non-tags, as shown in Figure 6.

**Prediction with one-class CNN model.** In the third step, a small CNN model is trained to determine whether the
segmented patch is a complete tag or not. Only patches with complete tags are kept and other patches are discarded. The CNN here is basically a one-class CNN problem, as it only tries to predict whether the input is a tag or not. It is easy to define what the class is with one-class classification. For example, the input segmented patches are judged to be tags if they contain a tag. However, it is difficult to define what is not a tag, as the environment could be different, and it is therefore unable to emulate all scenarios that are not tags. In our case, tags occluded by a large portion are considered as non-tag. To get the training data for this model, the samples of tags are generated via image transformation such as affine transformation, while randomly sampled environmental images are used for non-tag class. Therefore, no manual labeling is involved.

**Tag classification**

Since tags are distributed in the building and are associated with their real positions, if the segmented tags can be correctly identified, then the location of the camera can be determined. Since each tag has its own special features (or a combination of features), an exact mapping between tags and IDs can be established. To identify each tag, a neural network with manually made features is used in Anwar et al. Those manually made features are specially designed for their tags and are not guaranteed to be the best features. To improve the performance, a VGG16-based CNN is trained to predict the ID of the segmented tag. The CNN model is able to automatically extract better features toward the target tasks than those manually made features. The configuration for the classification-based CNN is shown on the left side of Figure 8. The VGG16 architecture contains 16 layers, and it follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end, it has two fully connected layers followed by a softmax for output. This model achieves 92.7% top-5 test accuracy in ImageNet, which is a data set of over 14 million images belonging to 1000 classes. It was one of the famous models submitted to ILSVRC-2014.
Distance and angle extraction

In the proposed system, the tag-camera distance and angle information is extracted from the segmented tag as illustrated in Figure 2. There is a certain relationship/mapping between the distance and the shape as well as the size of the tag that appears in the photo when the photo is taken from different angles. This mapping is a many-to-one mapping, mathematically written as

\[ d = f(\text{tag}_d) \quad (1a) \]
\[ d \neq f(\text{tag}_d'), d' \neq d \quad (1b) \]

where \( d \) is the real tag-camera distance and \( \text{tag}_d \) represents the tags taken at distance \( d \). Note that \( d \) is not related to the angle \( \theta \) as shown in Figure 3.

An example of such mapping is shown in Figure 9, where the images of the tags at the top are taken from 3 m away while the bottom ones are taken 7 m away at different angles.

Similarly, there exists a relationship/mapping between the angle \( \theta \) and the tag shape in the photo as can be seen in Figure 9. However, the shape of the tag in the image is affected not only by \( \theta \) but also by the rotation angle of the phone, \( \beta \), as shown in Figure 10. The mapping function between \( \theta \) and \( \beta \) may be described as

\[ \theta = f(\text{tag}_\theta, \beta) \quad (2a) \]
\[ \theta \neq f(\text{tag}_\theta', \beta'), \theta \neq \theta', \beta \neq \beta' \quad (2b) \]

where \( \text{tag}_\theta \) represents the tag taken at the angle of \( \theta \).

This effect is caused by a combination of different distortions introduced by the lens, and will be much worse with wide-angle lenses. So, this distortion should be considered in both distance and angle extraction. Although it can be estimated, the distortion comes from a combination of many different sources and is highly non-linear and complex. Thus, the correction is at best an approximation with few parameters and the errors are unevenly distributed over the image. The error of the corrected image with estimated distortions will affect the accuracy of localization since it is sensitive to the transformed tag shape in the image. When
the camera is rotated in different angles, the tag will appear in different locations in the image, as shown in Figure 11. Also, the mapping between the rotation angle $\beta$ and the location of the tag in the image is a one-to-one mapping expressed as

$$\beta = f(\text{tag}_l)$$  \hspace{1cm} (3)

where the $\text{tag}_l$ represents the location of tag in the image. Therefore, we have

$$\theta = f(\text{tag}_{\theta_l}, \text{tag}_l)$$  \hspace{1cm} (4a)

$$\theta \neq f(\text{tag}_{\theta_l'}, \text{tag}_l), \theta \neq \theta', \text{tag}_l \neq \text{tag}_l'$$  \hspace{1cm} (4b)

Instead of using distortion estimation and PE algorithms to estimate the location, CNN models are used. CNN models are able to learn the highly non-linear mappings directly from the original tag images to the final 2D position of the camera through training examples. Within these mappings, all distortions introduced by lenses and/or cameras have already been represented, avoiding the error of distortion estimation. This is also applicable to the distortions with wide-angle lenses.

The distortion is location dependent; for example, the barrel distortion along the border of an image is more severe than that at the middle. However, correction is done on the full-size image, which requires additional computation. For CNN models, the full-size images will greatly increase the computation if they are used as inputs. Instead of using full-size images as the input, to clearly tell the CNN the location of the tag in the image, we manually make an image, as shown in the left of Figure 12, where some black points in a blank (white color) image are used to represent the location of the tag in the image. The size of the manually made image is shrunk to the same size of the segmented tag,

![Figure 10. Camera rotation. Left: without rotation. Right: with rotation $\beta$.](image10.png)

![Figure 11. An example for rotating the phone horizontally when fixing the location of the phone and the tag on the wall.](image11.png)

![Figure 12. Manually made image for CNN. Left: an example for a manually made, black-and-white image to represent the location of the tag in the photo. Right: concatenate the manually made image and the segmented tag as the input to CNN.](image12.png)
so that they can be concatenated and passed to the CNN. This manually made image could be done easily since after tag segmentation, since the location of the tag in the photo is known.

**Location estimation**

With the tag ID, distance, and angle information, the location of the camera could be estimated using the approach shown in Figure 3. Note that, if more than one tag appears in a photo, the accuracy of location estimation will improve.

**Experiments and results**

**Experimental setup**

Experiments were conducted in a lab to implement the processes described in Figure 1. Tags as shown in Figure 5 with a size of 19 cm × 19 cm were printed in black-and-white. Ten different tags were used to test the tag segmentation and tag classification. Then, some tags were attached on the wall. About 1000 photos were taken with an iPhone for each tag from different distances, angles, and manually changed lighting conditions. To speed up the photo-taking process, the burst function of the iPhone was used, which resulted in severely blurred images.

The size and shape of the tag in the photo vary, and a square shape was used to segment the tag from the photos. Since the segmented images vary in size, they were padded with some values (e.g. 128) to make them the same dimension (e.g. 298 × 298 pixels), as shown at the bottom of Figure 9, where the small dark square is the segmented image and the outer gray square is after padding.

**Data set**

The data set contained about 10,000 photos for 10 tags. After discarding about 500 bad images that could not be recognized, the rest were segmented using the proposed methods. In this segmentation method, the main feature is the common square shape feature composed of small circles. There were about 9100 photos that could be correctly segmented; most of the failed cases were the severely blurred photos. The segmented tags were then labeled with their own IDs for CNN. To distinguish the different IDs, the different combination of small shapes inside the common square shape is used as the features as the inputs to the classification-based CNN.

**Training**

The classification-based CNN, as shown in Figure 8 (left), was trained with 80% of the segmented tags and validated on the remaining 20%. The validation accuracy was around 98%, which could be improved further with larger training sets and with better diversity.

To train the regression-based CNN for distance and angle estimation, we conducted an experiment as shown in Figure 13 in a lab, where there were 117 training positions and 46 testing positions. Due to the space limitation, the maximal distance was around 8 m. For each training and testing position, around 300 photos were taken.

![Figure 13](image-url)
taken with the burst function of an iPhone. To mimic the environmental lighting changes, the lighting density was manually changed during the data collection process. Each tag in the photo was segmented and labeled with its real distance and angle. Note that the distance and angle were normalized to within the range of [0,1] to facilitate the learning. The distance was normalized by 10 m; for example, an actual 5 m of distance was normalized to be 0.5. The angle was normalized by $\frac{180}{\pi}$ (i.e. the $90^\circ$ was normalized to be 0.5). All training data (here they refer to the segmented tags) were normalized with the mean and variance of the data set, and so did for testing data.

Both distance and angle were trained with the same regression-based CNN, as shown at the right of Figure 8, and with the manually made image, as shown in Figure 12. Note that the distance and angle were trained separately for the 2D case (i.e. only the $x$- and $y$-coordinates were considered with camera kept at the same height when data were acquired).

**Results**

Test results for distance and angle are shown in Figures 14 and 15, respectively. The mean of the distance estimation error is low, mostly less than 10 cm, and the error for various distances is consistent. The standard deviation is also low in the whole range, but the value slightly increased with distance. The mean of the angle estimation error is less than $6^\circ$, with about half less than $3^\circ$. It is found that the angle error increases when the camera is around $90^\circ$, likely because in these areas the change of the shape of the tag is too small and is thus difficult to recognize by the CNN.

![Figure 14. Distance extraction results.](image1)

![Figure 15. Angle extraction results.](image2)
Localization experiments with one and two tags were conducted. During the experiments, the lighting change was also applied manually. In the one-tag case, each photo contains one tag, and the camera’s position is the intersection of the semi-circle and the line, as illustrated in Figure 3. The tag position, camera’s real positions, and the estimated positions are shown in Figure 16. At each testing location, around 100 photos were taken. The average location error is found to be about 1 m.

In this case, the average location error is found to be less than 20 cm.

Compared with the ML-based method with hand designed features, the above testing results show that the proposed end-to-end learning system can also achieve reasonable positional accuracy with automatic feature learning. And the performance is relatively consistent with distance, unlike the one in Anwar et al. with PE-based method, whose accuracy drops with the distance. In addition, the testing results considered the lighting changes, which the other methods did not.

**Analysis of error factors**

As described by equations (1) and (4), the positioning accuracy is a function of both the distance $d$ and the angle $\theta$. Possible causes for the errors observed in the experiments above are discussed next.

First, training data were limited. Both $d$ and $\theta$ are continuous quantities. However, training data were collected at discrete locations. For example, $d$ was provided at 2, 3 m, and so on, and $\theta$ was sampled with a 15° step from 30° to 150°. Increasing spatial sampling frequency for $d$ and $\theta$ for training will improve the performance.

Second, since the angle is related to the shape of the tag in the image instead of the size, it is possible to up-sample the tag when the tag is small (or when the camera is far away from the tag) to increase the image resolution so that CNN performs better.

Third, increasing the sample data ($d$ and $\theta$) measurement accuracy will increase performance. The errors in Figures 14 and 15 are inconsistent, especially those caused by the angle. For example, the mean of the absolute error for $\theta$ from 40° to 80° jumps between 0° and around 4°, which is likely caused by measurement errors.

**Discussion**

**Resolving the issue of high computation needs**

The proposed system is DL-based, which requires much more computational power than traditional PE algorithms. However, optimization methods from recent developments in DL community could help achieve real-time performance even on mobile or embedded devices. The proposed system consists of three DL models (for tag identification, and distance and angle extraction), which take up the majority of the computations. There are several ways to optimize it. First, the latest MobileNet and GhostNet architectures could save up to 100 times of the computations while maintaining a similar performance when compared to the VGG-16 network used in the experiments. Second, since the three DL models use the same inputs, they can be merged into a single model (requiring the computation of only one model) by sharing the same weights.
Building a standard flow for quick deployment. As discussed in section “Introduction,” image-tag-based systems using PE algorithms have major problems that prevent them from being widely deployed: tag density must be high to maintain a good accuracy, and each system designed is specific to a set of pre-designed tags and lenses/cameras; if the tags and/or lenses/cameras have changed, then domain experts are required to redesign the system to adapt to the new changes.

The proposed system overcomes these problems, and facilitates a design with a standard flow to achieve excellent performance with much fewer tags and much greater adaptability to new changes without requiring knowledge of domain experts. First, using wide-angle lenses will dramatically reduce tag density. However, this will introduce in barrel and potentially other distortions for which PE algorithms are ineffective to deal with, as discussed in section “Introduction.” The ML-based method simply ignores such distortions. The proposed method takes into account all distortions by mapping the image tag to the final positions using end-to-end learning, where the mapping relationship contains all effects including distortions, as shown in section “Location estimation.” Thus, it naturally accommodates the use of wide-angle lenses.

Second, the proposed method involves steps of data collection, tag detection, tag identification, distance, and angle extraction. Data collection as illustrated in Figure 13 a standard process that can be done by anyone. Tag identification and distance and angle extraction processes are all DL-based, which requires a large amount of labeled data for the DL model training. With a careful design, however, this labor-intensive work can be avoided. After data collection as shown in Figure 13, all captured images or videos have already been associated with labels including tag ID, distance, and angle information. Through template matching style detection, which does not need labeling, cropped tags are associated with corresponding labels automatically. The remaining processes are for training or fine-tuning the DL models, which can be automated with new training data once the DL architectures are defined and the training hyper parameters are set. Therefore, via such a standard flow, the proposed system can be quickly and widely deployed without requiring domain expertise.

In addition to the standard flow mentioned above, to make the system adaptable to other different cameras, the in-device software will be required to capture and control the lens parameters. And these parameters will be part of the input to the algorithm to adapt the changes.

Conclusion

A new image-tag-based indoor localization method has been proposed in this article. With uniquely designed tags that are trained and can be populated in any space where positioning take place, we train several CNNs to classify the tag ID, and to estimate the distance as well as angle information with the segmented tag in the image taken by a camera. The position of the camera is then determined from the extracted distance and angle. The tags need to be trained only once and then can be used anywhere, which makes deployment extremely simple. Once a large number of tags are designed and trained, coverage becomes highly scalable and blind spots will no longer be an issue. The system is also robust to environmental changes and the localization accuracy could increase drastically by increasing tag density. Experiments in a lab have achieved less than 20 cm of localization errors with two tags per image.

Author contributions

Z.S. and H.L. contributed to the conceptualization. Z.S., R.L., and M.A. contributed to the methodology. Z.S. and R.L. contributed to the software. H.L., Z.S., and M.A. contributed to the writing—original draft preparation. R.L. and A.A.-H. contributed to the writing—review and editing. H.L. contributed to the supervision. M.A. and A.A.-H. contributed to the funding acquisition.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: The authors acknowledge the Deanship of Scientific Research at King Faisal University for the financial support under Ra’ed Track (Grant No. 207002).

ORCID iD

Mohammed Alarfaj https://orcid.org/0000-0001-6089-9319

References

1. Lee N, Ahn S and Han D. AMID: accurate magnetic indoor localization using deep learning. Sensors 2018; 18: 1598.
2. Badrinarayanan V, Kendall A and Cipolla R. SegNet: a deep convolutional encoder-decoder architecture for image segmentation, 2016, https://arxiv.org/pdf/ 1511.00561.pdf
3. Mautz R. Indoor positioning technologies. Doctoral Dissertation, Institute of Geodesy and Photogrammetry, ETH Zurich, Zurich, 2012.
4. Sattler T, Leibe B and Kobbelt L. Fast image-based localization using direct 2D-to-3D matching. In: Proceedings of the 2011 international conference on computer vision (ICCV’11), Barcelona, 6–13 November 2011, pp.667–674. New York: IEEE.
5. Sinha D, Ahmed MT and Greenspan M. Image retrieval using landmark indexing for indoor navigation. In: Proceedings of the 2014 Canadian conference on computer and robot vision (CCR’14), Montreal, QC, Canada, 6–9 May 2014, pp.63–70. New York: IEEE.
6. Huang Y, Wang H, Zhan K, et al. Image-based localization for indoor environment using mobile phone. Int Soc Photogramm 2015; 4: 211–215.
7. Walch F, Hazirbas C, Leal-Taixe L, et al. Image-based localization using LSTMs for structured feature correlation. In: Proceedings of the 2017 IEEE international conference on computer vision (ICCV), Venice, 22–29 October 2017, pp.627–637. New York: IEEE.
8. Huang JY, Lee SH and Tsai CH. A fast image matching technique for the panoramic-based localization. In: Proceedings of the 2016 IEEE/ACIS 15th international conference on computer and information science (ICIS), Okayama, Japan, 26–29 June 2016, pp.1–6. New York: IEEE.
9. Ravi N, Shankar P, Frankel A, et al. Indoor localization using camera phones. In: Proceedings of the 7th IEEE workshop on mobile computing systems and applications (WMCSA’06 supplement), Orcas Island, WA, 1 August 2005. New York: IEEE.
10. Bay H, Tuytelaars T and Van Gool L. Speeded up robust features. In: Proceedings of the European conference on computer vision (ECCV’06), Graz, 7–13 May 2006, vol. 4, pp.404–417. Berlin; Heidelberg: Springer.
11. Mulloni A, Wagner D, Barakonyi I, et al. Indoor positioning and navigation with camera phones. IEEE Pervas Comput 2009; 8(2): 22–31.
12. Babinec A, Jurišica L, Hubinský P, et al. Visual localization of mobile robot using artificial markers. Procedia Engineer 2014; 96: 1–9.
13. Debski A, Grajewski W, Zaborowski W, et al. Open-source localization device for indoor mobile robots. Procedia Comput Sci 2015; 76: 139–146.
14. Liao J-K and Tsai G-J. The development of an artificial neural networks aided image localization scheme for indoor navigation applications with floor plans built by multi-platform mobile mapping systems. In: Proceedings of the 30th international technical meeting of the satellite division of the institute of navigation (ION GNSS +, 2017). Portland, OR, 25–29 September 2017, pp.3010–3027. Portland, Oregon: Oregon Convention Center.
15. Xing B, Zhu Q, Pan F, et al. Marker-based multi-sensor fusion indoor localization system for micro air vehicles. Sensors 2018; 18(6): 1706.
16. Li Y, Zhu S, Yu Y, et al. An improved graph-based visual localization system for indoor mobile robot using newly designed markers. Int J Adv Robot Syst. Epub ahead of print 16 April 2018. DOI: 10.1177/1729881418769191.
17. Gang H-S and Pyun J-Y. A smartphone indoor positioning system using hybrid localization technology. Energies 2019; 12: 3702.
18. Kim G and Petriu EM. Fiducial marker indoor localization with artificial neural network. In: Proceedings of the 2010 IEEE/ASME international conference on advanced intelligent mechatronics, Montreal, QC, Canada, 6–9 July 2010. New York: IEEE.
19. Anwar Q, Malik A and Thörnberg B. Design of coded reference labels for indoor optical navigation using monocular camera. In: Proceedings of the international conference on indoor positioning and indoor navigation, Montbéliard, 28–31 October 2013, pp.112–120. New York: IEEE.
20. Wu RQ, Liu J, Chen W, et al. Adaptive wide-lens distortion correction based on piecewise polynomial optimization. Procedia Comput Sci 2019; 154: 573–580.
21. Zou Y and Liu H. An efficient NLOS errors mitigation algorithm for TOA-based localization. Sensors 2020; 20: 1403.
22. Su Z, Shao G and Liu H. Semidefinite programming for NLOS error mitigation in TDOA localization. IEEE Commun Lett 2018; 22(7): 1430–1433.
23. Zou Y, Liu H, Xie W, et al. Semidefinite programming methods for alleviating sensor position error in TDOA localization. IEEE Access 2017; 5: 23111–23120.
24. Barral V, Escudero CJ, Garcia-Naya JA, et al. Environmental cross-validation of NLOS machine learning classification/mitigation with low-cost UWB positioning systems. Sensors 2019; 19: 5438.
25. Tian X, Wei G, Wang J, et al. A localization and tracking approach in NLOS environment based on distance and angle probability model. Sensors 2019; 19: 4438.
26. Qiao T and Liu H. Improved least median of squares localization: technical challenges and methods. In: Proceedings of the 2010 IEEE international conference on advanced intelligent mechatronics, Montreal, QC, Canada, 6–9 July 2010. New York: IEEE.
27. Qiao T and Liu H. An improved method of moments estimator for TOA based localization. IEEE Commun Lett 2013; 17(7): 1321–1324.
28. Qiao T, Redfield S, Abbasi A, et al. Robust coarse position estimation for TDOA localization. IEEE Wirel Commun 2013; 20(6): 623–626.
29. Zou Y, Liu H and Wan Q. Joint synchronization and localization in wireless sensor networks using semidefinite programming. IEEE Internet Things J 2018; 5(1): 199–205.
30. Ye R, Redfield S and Liu H. High-precision indoor UWB localization: technical challenges and method. In: Proceedings of the 2010 IEEE international conference on ultra-wideband, Nanjing, China, 20–23 September 2010. New York: IEEE.
31. Arias S. Visual tag recognition for indoor positioning. Master’s Thesis, KTH Royal Institute of Technology, Stockholm, 2013.
32. Zhong X, Zhou Y and Liu H. Design and recognition of artificial landmarks for reliable indoor self-localization of mobile robots. Int J Adv Robot Syst 2017; 14(1): 1–13.
33. Chang Y-H, Chu C-H and Chen M-S. A general scheme for extracting QR code from a non-uniform background in camera phones and applications. In: Proceedings of the 9th IEEE international symposium on multimedia (ISM’2007), Taichung, Taiwan, 10–12 December 2007, pp.123–130. New York: IEEE.

34. Fiala M. Designing highly reliable fiducial markers. IEEE T Pattern Anal 2010; 32(7): 1317–1324.

35. Rekimoto J and Ayatsuka Y. CyberCode: designing augmented reality environments with visual tags. In: Proceedings of DARE 2000 on designing augmented reality environments (DARE’2000), Elsinore, 12–14 April 2000, pp.1–10. New York: ACM.

36. Liu W, Anguelov D, Erhan D, et al. SSD: single shot multibox detector. In: Proceedings of the European conference on computer vision (ECCV), Amsterdam, 8–16 October 2016, pp.21–37. Cham: Springer.

37. Redmon J and Farhadi A. Yolo9000: better, faster, stronger. In: Proceedings of the 2017 IEEE conference on computer vision and pattern recognition (CVPR), Honolulu, HI, 21–26 July 2017, pp.6517–6525. New York: IEEE.

38. Bay H, Ess A, Tuytelaars T, et al. Speeded-up robust features (SURF). Comput Vis Image Und 2008; 3(2): 346–359.

39. Lowe DG. Distinctive image features from scale-invariant keypoints. Int J Comput Vision 2004; 60(2): 91–110.

40. Xu D and Tian Y. A comprehensive survey of clustering algorithms. Ann Data Sci 2015; 2(2): 165–193.

41. Tax D. One-class classification. PhD Thesis, Delft University of Technology, Delft, 2001.

42. Simonyan K and Zisserman A. Very deep convolutional networks for large-scale image recognition. In: Proceedings of the international conference on learning representations (ICLR’2015), San Diego, CA, 7–9 May 2015, Conference Track Proceedings, arXiv 1409.1556.

43. Sandler M, Howard A, Zhu M, et al. MobileNetV2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), Salt Lake City, UT, 18–23 June 2018, pp.4510–4520. New York: IEEE.

44. Han K, Wang Y, Tian Q, et al. GhostNet: more features from cheap operations. In: Proceedings of the 2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR), Seattle, WA, 13–19 June 2020, pp.1580–1589. New York: IEEE.

45. Zhang Z, Luo P, Loy CC, et al. Facial landmark detection by deep multi-task learning. In: Proceedings of the European conference on computer vision (ECCV), Zurich, 6–12 September 2014, pp.94–108. Cham: Springer.

46. TensorRT, https://developer.nvidia.com/tensorrt#:~:text=NVIDIA5