Automated Virtual Navigation and Monocular Localization of Indoor Spaces from Videos

Qiong Wu Ambrose Li
HERE Technologies
210-4350 Still Creek Dr. Burnaby, BC Canada V5C 0G5
{qiong.wu, ambrose.li}@here.com

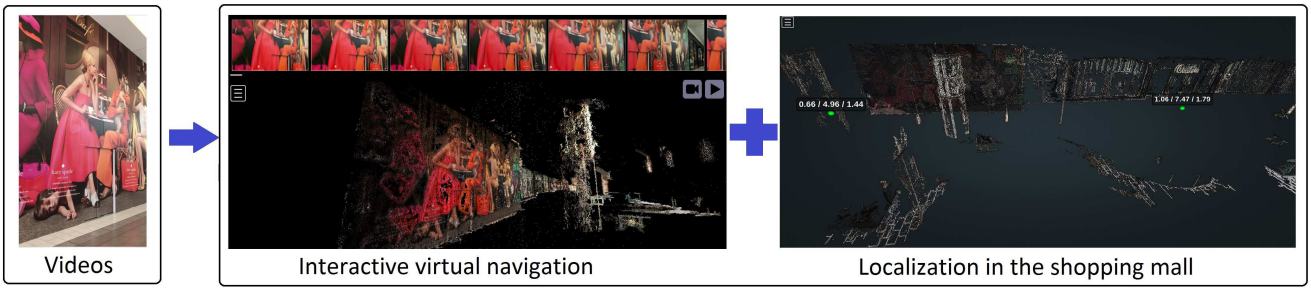


Figure 1: Given videos of an environment, our system can automatically process and make two services available: 3D interactive virtual navigation and image-based localization.

Abstract

3D virtual navigation and localization in large indoor spaces (i.e., shopping malls and offices) are usually two separate studied problems. In this paper, we propose an automated framework to publish both 3D virtual navigation and monocular localization services that only require videos (or burst of images) of the environment as input. The framework can unify two problems as one because the collected data are highly utilized for both problems, 3D visual model reconstruction and training data for monocular localization. The power of our approach is that it does not need any human label data and instead automates the process of two separate services based on raw video (or burst of images) data captured by a common mobile device. We build a prototype system that publishes both virtual navigation and localization services for a shopping mall using raw video (or burst of images) data as inputs. Two web applications are developed utilizing two services. One allows navigation in 3D following the original video traces, and user can also stop at any time to explore in 3D space. One allows a user to acquire his/her location by uploading an image of the venue. Because of low barrier of data acquirement, this makes our system widely applicable to a variety of domains and significantly reduces service cost for potential customers.

1. Introduction

3D visual models of indoor environments are useful in applications such as navigation, virtual reality and entertainment. It can provide detailed knowledge about the environment as well as contextual information for users and allow their interactions with the environment. Monocular localization is a relatively new rising area of study. Since global positioning system (GPS) typically cannot communicate with the satellites inside the buildings, indoor localization and navigation is still an open problem which has potential huge impact on many commercial and public services. Both fields have wide applications and are well studied. However, most technologies for one field are developed independent of the other as they are considered two separate problems. As an example, monocular localization does not require 3D visual model and is, therefore, unrelated to virtual navigation. The result of this disconnection between the two problems is that the production pipeline for virtual navigation cannot be utilized for monocular localization. In short, a hybrid technology that can achieve both virtual navigation and monocular localization at the same time does not exist yet.

Virtual navigation requires 3D models. Most early technologies for building accurate 3D models require heavy-duty laser scanners, which are not easily accessible to average users. Second tier of 3D reconstruction technology uses less expensive depth cameras such as Kinect [5]. Vision-based 3D modeling is the third tier and most cost effective method. Photo Thourism has shown 3D structures can be recovered from photos. However, all technologies for 3D modeling do not consider achieving the goal of monocular localization.

Early methods for indoor localization are sensor-based approaches, which require infrastructure installation (e.g., WiFi access points or beacons with known positions). Those sensors are either densely distributed within the scene or pre-assume initial absolute locations [10, 2]. This implied heavy deployment cost and labor requirements at the venue to be mapped. They also have operational limitations due to them being battery powered. With the focus shifting to minimizing infrastructure cost without compromising substantially on accuracy, there have been many attempts at vision-based localization. Many vision-based localization approaches require the preparation of a database of images with their corresponding location for the venue. Such a database usually contains only images but not a 3D model of the venue. Localization then involves indexing in the dataset by matching the visual appearance and/or geometry. Other vision-based localization methods [18] require significant manual labeling work to generate training data.

The key idea behind our work is built on the following two observations. First, if vision-based localization approaches are based on a learning method that uses images as training data, then the 3D visual model for virtual navigation should be able to reuse those image data. Second, 3D visual model contains information about how a geo-location co-relates with an image, which should be useful for vision-based localization. Achieving both goals and providing two services at the same time are useful in many scenarios. Suppose you get lost somewhere in a shopping mall and have a hard time to describe where you are, and you need to go to another store. Instead of finding shops name and looking them up on a directory map, the easiest way to locate yourself is to snap a photo of the store nearby. Once you are localized, a path may be planed with the desired destination store's name. The navigation could then be assisted by the virtual navigation.

Following this intuition, we present an automated framework that publishes both monocular localization and 3D virtual navigation services with simple video inputs. The framework highly utilizes the pipeline of building 3D visual model for monocular localization, and reuses the data from the processing results of building 3D model as the training data for monocular localization. The main contributions of our work are three folds. First, we present an automation framework that publishes both virtual navigation and monocular localization services with videos (or burst of images) as inputs. Second, we share a new dataset for a part of the shopping mall. Third, as an alternative data collection method, we present a tool and method that captures a burst of images with indoor position geotags, and transform low accuracy discrete geotags into high accuracy continuous geotags. Our data is publicly available at `https://goo.gl/j2KUrc`.

2. Related Work

Since one of the main goals for our work is to solve indoor localization, we here mainly review the related work in this field.

The conventional indoor localization focuses mainly on location accuracy and involves the use of custom sensors [4, 19] such as WiFi access points and Bluetooth iBeacons. It requires deployment of anchor nodes in the environment and sometimes even sensors for users. For example, a WiFi-based positioning system measures the intensity of the received signal from the surrounding WiFi access points for which the location is known. This implied heavy deployment cost and labour requirements at the venue to be mapped. The maintenance of geolocalized WiFi dataset also requires maintenance to prevent being out-of-date. Moreover, such localization accuracy may be varied depending on the changes in signal strength, and only performs well in the area with a sufficient number of sensors to enable triangulation calculation.

With the focus shifting to minimizing infrastructure cost without comprising substantially on accuracy, there have been many attempts at vision-based localisation. Approaches of this kind mainly fall into three categories: metric based, appearance based, and additional cue based. Simultaneous localization and mapping (SLAM) [6, 8] and structure-from-motion (SfM) [1, 14] are metric based. They are mainly used for mobile robot localization. Camera's pose are calculated based on the relative movement to the previous position or the collection of images. Appearance based localization provides a coarse estimate by comparing visual features of the query image against the scene described by a limited number of images with location information. For example, using SIFT features [12] in a bag of words approach has been proposed to probabilistically classify the query image. Deep learning based approaches which learn visual features automatically also belong to this category. For example, Convnets [15] classifies a scene into one of location labels and PoseNet [11] regresses locations to localize the camera. Additional cue based approaches [7, 3, 18] mainly incorporate the map data as an additional cue into the localization framework. However, usually those data requires heavy manual labeling labour in order to be useful for the system. For example, [18] uses Amazon Me-

chanical Turk (AMT) to label 3D polygons enclosing shop's names in the images.

The closest to our work is PoseNet [11] which use SfM [20] as an offline tool to prepare training data and [9] to generate a dense visualisation mainly for visualizing the re-localisation results purpose. Our work takes it a step further, re-using the data from 3D visual model and achieving virtual navigation at the same time. In addition, we also develop the localization training part in Tensorflow framework and compare with PoseNet Caffe framework on the King's College [1] dataset and the result shows our training is superior.

3. System Overview

In this paper, we propose an automation framework that enables two web-based services simply with videos as inputs: 3D visualization and monocular localization. Figure 2 illustrates the automation framework. It takes videos of a venue as input and provides two web services for the venue as output. The core components of the framework contain two production engines and one process monitor engine. The 3D reconstruction engine mainly contains SfM methods to generate 3D visual model and image data labeled with camera pose information. The localization training engine takes labeled data prepared by the 3D reconstruction engine as the training data to train a recognition model using machine learning algorithms. Subsequently, the process monitor engine coordinates the production, monitors the process, and publishes web services once required data is available.

Below we describe the implementation details of two production engines and the process monitor engine.

3.1. 3D reconstruction engine

It implements a series of SfM algorithms [20] that take a set of images as input and computes a 3D point cloud, back-calculated camera positions in 3D space for all images, and a 3D dense visual model as output. The images could come from sampled video frames or captured by our in-house developed tool. SfM algorithms mainly include feature extraction and matching, back-calculation of 3D position for each pixel in images, minimal solvers, and camera pose including position and orientation for each image. When geo position (e.g., indoor position) measurements are available for images (e.g. captured by in-house developed tool), then the 3D model can be registered with the corresponding geo-location.

For this part, we compared two different SfM production pipelines, vsfm [9] and OpenMVG [13]. After experiments, we discover that OpenMVG is superior to vsfm in terms of 3D reconstruction speed, accuracy, ease of code extension and automation integration. After comparison, we employed OpenMVG as part of 3D reconstruction engine and integrated into our system prototype. We present the comparison results in the experiment part.

[1]The dataset is available on PoseNet project webpage: mi.eng.cam.ac.uk/projects/relocalisation/

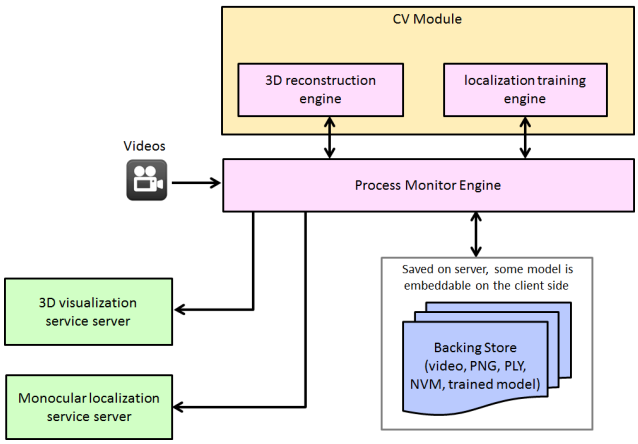


Figure 2: Detailed system overview: the system accepts videos as inputs and provides two services: 3D visualization and image-based localization. The core components of the framework contains two production pipelines, 3D reconstruction engine and localization training engine

3.2. Localization training engine

This part could be any machine learning method that takes a set of labeled data (e.g. our prototype uses images labeled with corresponding camera pose information) as input and trains a recognition model for monocular localization service.

In our prototype, we use Tensorflow framework to develop the model for deep regression of camera pose proposed in PoseNet [11]. Unlike PoseNet which uses Inception v1 [16] as a basis for developing pose regression network, we employed Inception v3 [17] as the basis. We modify the Inception v3 similar to the way PoseNet proposed as follows:

- Because Inception v3 has only one auxiliary classifier, we replace all two softmax classifiers with affine regressors. The sofmax layers were removed, and the final fully connected layer was changed to 7-dimensional pose vector representing position (3) and orientation (4).
- Insert fully connected layer before the regressors of feature size 2048 both on the final classifier and auxiliary classifier.

Because we use video data which was taken in portrait mode as our training data, we only rescaled the input im-

age to 224×224 resolution without random cropping. This could help us avoid training with images cropped to contain mainly floors or ceilings of the shopping mall. At test time, we evaluate an image also by rescaling it to 224×224 pixel size.

Like many deep learning applications which apply "transfer learning" for training classifiers. We applied transfer learning by training with Places205[2] dataset first. Because there is not a pre-trained model available for Inception v3, we trained it with Places205 dataset from scratch. Our submission shows 0.5961 top 1 accuracy and 0.8837 top 5 accuracy. The model is then further trained with our own shopping mall data.

3.3. The process monitor engine

This engine controls the processing order, preparation of data for the production pipeline and web services, and serialization of data to backend server. The process is controlled so that localization training engine will be synchronized with the availability of input data from the 3D reconstruction process. There are four kinds of controls in the process monitor engine:

- Preparation of input data for 3D reconstruction engine: This is a control script that extracts image frames from video inputs at a predefined frequency (e.g. two frames per second). Once the 3D reconstruction process is completed, the script serializes new information such as point cloud, camera positions, and 3D model to the backend data server.

- Preparation of data for 3D visualization service: When the 3D visual model is available, this script converts model into a format that can be visualized on web browser. Potree is applied for visualizing models on web browser.

- Preparation of input data for localization training engine: This script prepares the training data for the localization training engine. The training data includes image frames extracted from videos, and a file describing the camera pose information, which includes data on the position and orientation for each image. The camera pose information is saved to the backend data server as part of the processing results from the 3D reconstruction engine. Once the training process has completed, the script also serializes new information, such as recognition model, to the server.

- Preparation of data for monocular localization service: When the recognition model is available, this script publishes an image-based localization service by loading the recognition model. The service outputs a predicted camera pose including position and orientation by giving an image of the venue.

[2]http://places.csail.mit.edu/downloadData.html

4. Dataset

We have two means of collecting data. One is capturing raw video data from a mobile phone camera. Another one is using an in-house developed tool which captures a burst of images with corresponding geotag information. Although we only use video data in our prototype, the later one which collects data with geolocation information may be more useful in many applications, such as aligning a visual model with the corresponding floor plan.

4.1. Video data

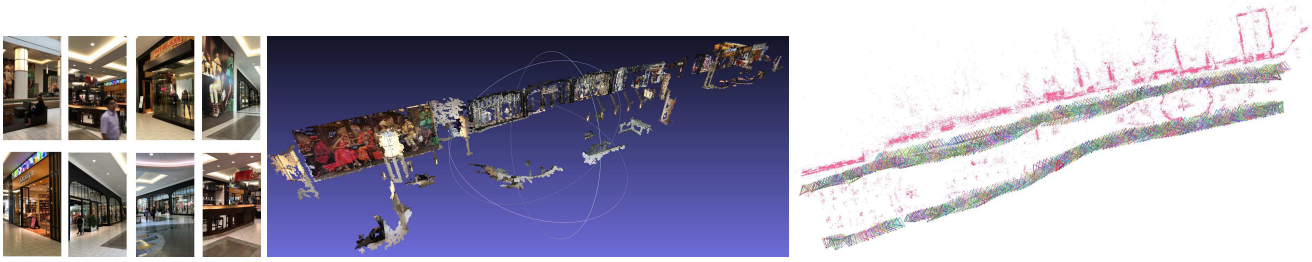
We collected a new dataset for a shopping mall named Metrotown in Vancouver. In order to capture variance of appearance, illumination, and lighting changes, we visited the mall in a few different times spanning from April to August in 2017. We recorded 12 video sequences for the same section of the shopping mall with an iPhone 7 device and took different paths in order to cover a complete view of the section. This video is then sub-sampled at 2Hz frequency to generate images which are used as input to the 3D reconstruction engine. In total there are 1882 images, and we split them into 1247 training images, and 635 testing images. Figure 3 illustrates sampled video frames, 3D visual model from SfM pipeline, and back-calculated camera pose for each sampled image. Our data is publicly available at `https://goo.gl/j2KUrc`.

4.2. Burst of images

As an alternative way of data collection, we also developed a tool for taking burst of images with geolocation information which is saved in image's geotag. The tool has features including setting capture rate and resolution, blurry image indicator that may indicate the user is moving too fast, and indoor position selection such as GPS, WiFi network, or radio maps. Figure 5 shows screen shots of the tool.

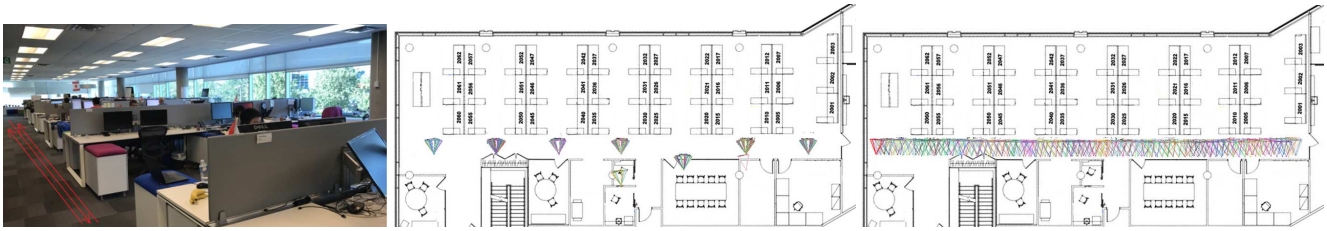
One problem we find while capturing images with this tool is that geotags generated through sensor technologies relying on Bluetooth beacons or RFID tags can be error prone due to limitations in the technology. For example, for three continuous walking paths along an indoor corridor, as shown in Figure 4a, geotags recorded via Bluetooth sensors create a radio map of discrete locations, as shown in Figure 4b. In order to correct erroneous geotags, we propose extrapolate geotags for images according to their virtual camera positions computed through SfM pipeline. This is a two-step process:

- Compute virtual camera position by SfM pipeline. In SfM pipeline, the first step is the extraction and match-



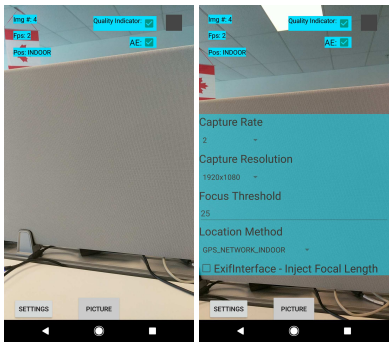
(a) Sample video frames **(b)** 3D model **(c)** Camera poses

Figure 3: Original video data and processed resulting data



(a) Office view **(b)** Collected Geotags are in discrete **(c)** Transformed Geotags are in continuous

Figure 4: Data collection by the burst image capturing tool. (a) Continuous images are taken at 2 frames/second using an internally developed image-capturing tool on an Android device held by the operator walking along the office corridor. The red lines in the image represent three different walking paths, and the arrow represents the direction of travel. (b) Captured images are grouped according to the recorded geotag, which shows their positions are in discrete rather than continuous. An image is represented by a tetrahedron. (c)After transformed, the camera positions are labeled with continuous Geotags.



(a) Preview screen **(b)** Settings screen

Figure 5: Screen shots of the function features developed in the tool.

ing of features within the images taken at the same venue or scene. This is done using a Scale Invariant Feature Transform (SIFT) algorithm. An image feature is an area of image texture containing patterns that is likely to be recognized in other images, such as logos and patterns with strong color contrast. The second step is matching these features across an image set. A sparse point cloud where each point in the image is back projected in a 3D space in addition to the back-calculated position and orientation of the virtual cameras for these images are computed.

- Extrapolate geotags for images according to their virtual camera positions. Once the continuous virtual camera positions are available, geotags of the selected images are used to calculate a transformation model between their virtual camera positions and geo locations. Different algorithms can be used to compute the transformation. For example, RANSAC can estimate a 3D similarity transformation $X' = S * R * X + T$. Once calculated, the transformation model is applied to the rest of geotags to correct their location errors. The resulting corrected geotags are illustrated in Figure 4c.

5. Experimental results

In this section, we present our experiment results on localization accuracy, and comparison of OpenMVG to vsfm. We also demonstrate two web services published using our prototype, virtual navigation and monocular localization.

5.1. Comparison of localization accuracy

We show that our implementation performs better than PoseNet on localization accuracy. We present our exper-

Table 1: Detaset details and results

Scene	# Frames Train/Test	Spatial Extent(m)	PoseNet	Ours
King's College	1220/343	140*40m	1.92m, 2.70°	1.17m, 2.25°
Metrotown (vsfm)	1247/635	41*10m	N/A	1.44m, 4.39°
Metrotown (openMVG)	1259/316	10*3m	N/A	0.29m, 1.53°

imental evaluation on King's College dataset[3] released by PoseNet first. As shown in Table 1, we achieve 1.17m, 2.25° compared to 1.92m, 2.70° from PoseNet at the sale of 140×40m spatial extent.

For the Metrotown dataset, we experiment with two different SfM pipelines, vsfm and openMVG. Because of different algorithms implementation, they achieve different reconstruction results and have different number of images in their final 3D visual model. As shown in Table 1, vsfm generates 1247/635 train/test dataset, reaching the scale at $40 * 10$m spatial extent. Using this dataset, our localization accuracy is 1.44m, 4.39°. OpenMVG generates 1259/316 train/test dataset, reaching the scale at $10 * 3$m spatial extent. Using this dataset, our localization accuracy is 0.29m, 1.53°. Taking into count that openMVG dataset is almost 4 times downsizing from vsfm dataset, our localization performs much better in terms of both camera position and orientation using openMVG dataset than vsfm dataset. This is because openMVG performs better on image matching etc than vsfm. We present more visual comparison in Section 5.4

5.2. Virtual Navigation Service

Utilizing the 3D visual model data, we publish a web-based virtual navigation service. As shown in Figure 6, top rows displays all video frames and it allows a users to navigate the space either following the traces of original video data or explore by oneself. User may optionally select the video frame from top row, then the image overlays on top of the visual model. User may also optionally move in 3D space to see how the camera is positioned and orientated in the 3D space for the selected video frame.

5.3. Monocular Localization Service

Utilizing the recognition model data, we publish a web-based monocular localization service. As shown in Figure 7, a user may get a predicted location by uploading an image of the scene. Here we demonstrate the performance of our system by using testing images captured by a mobile device, e.g. Android phone, different from the one captured the original video data. Given the testing images taken at different portrait mode and resolution, our system still has decent prediction. On average, the prediction takes about 0.13sec for processing a query image.

[3]The dataset is available on PoseNet project webpage: mi.eng.cam.ac.uk/projects/relocalisation/

Figure 6: Virtual navigation. Left: virtual navigation; right: overlaying of video frames on top of the 3D visual model. User can explore by themselves in 3D space.

5.4. Comparison of OpenMVG to vsfm

Here we demonstrate different 3D reconstruction results using OpenMVG and vsfm. Figure 8 shows 3D reconstruction and camera poses calculation from OpenMVG and vsfm. As one can see, vsfm has more matched images and camera poses in the reconstruction results. That is why more details are shown in the 3D reconstruction results. However, because of lower matching accuracy, openMVG results has a clearer model. Figure 9 shows a zoom-in view on the visual model details. As one can see, reconstruction details such as store name logos look more clean and in better accuracy in OpenMVG results, although vsfm has more area (e.g. around logos) successfully reconstructed by vsfm.

6. Conclusions

Although our work is not about a new method in either 3D reconstruction or monocular localization, we pro-

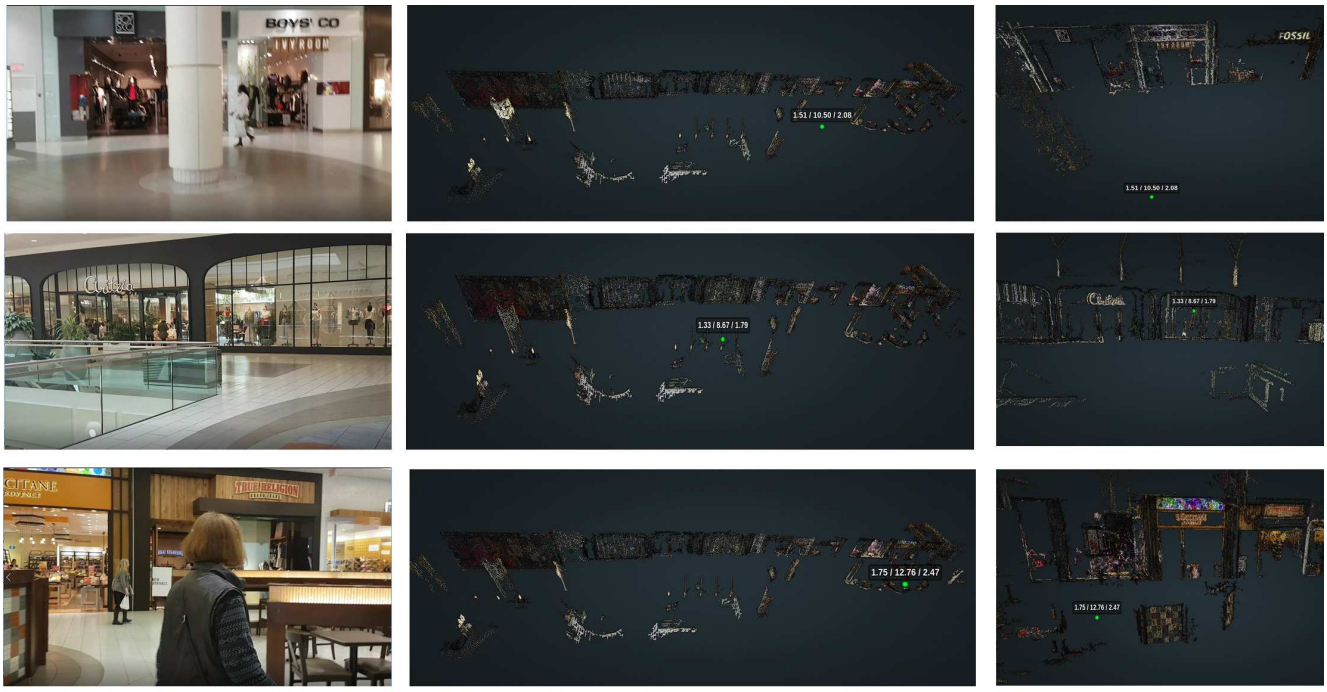


Figure 7: Localization results using images captured by an Android device at a completely different landscape mode and resolution. Left: test image; middle: predicated location; right: zoom in view on the location prediction

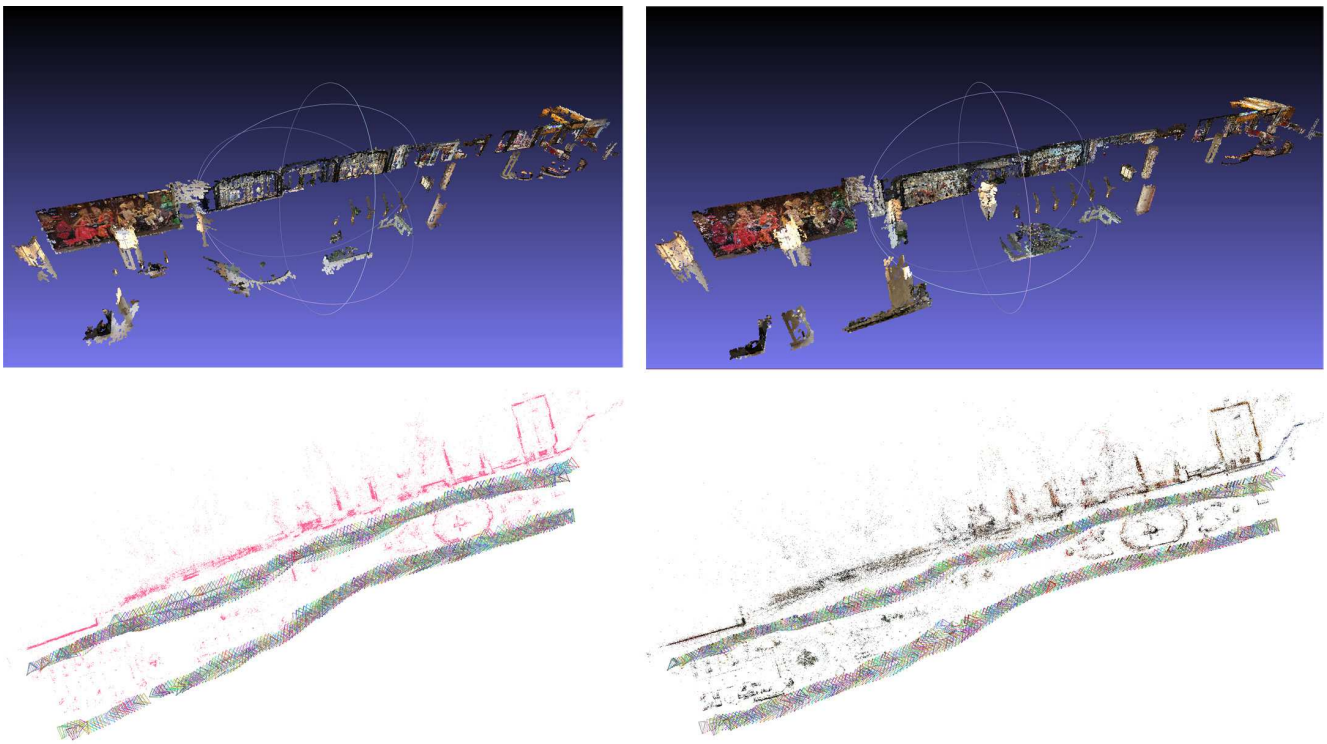


Figure 8: Comparison of 3D model and back-calculated camera positions. Left: OpenMVG results; Right: vsfm results.

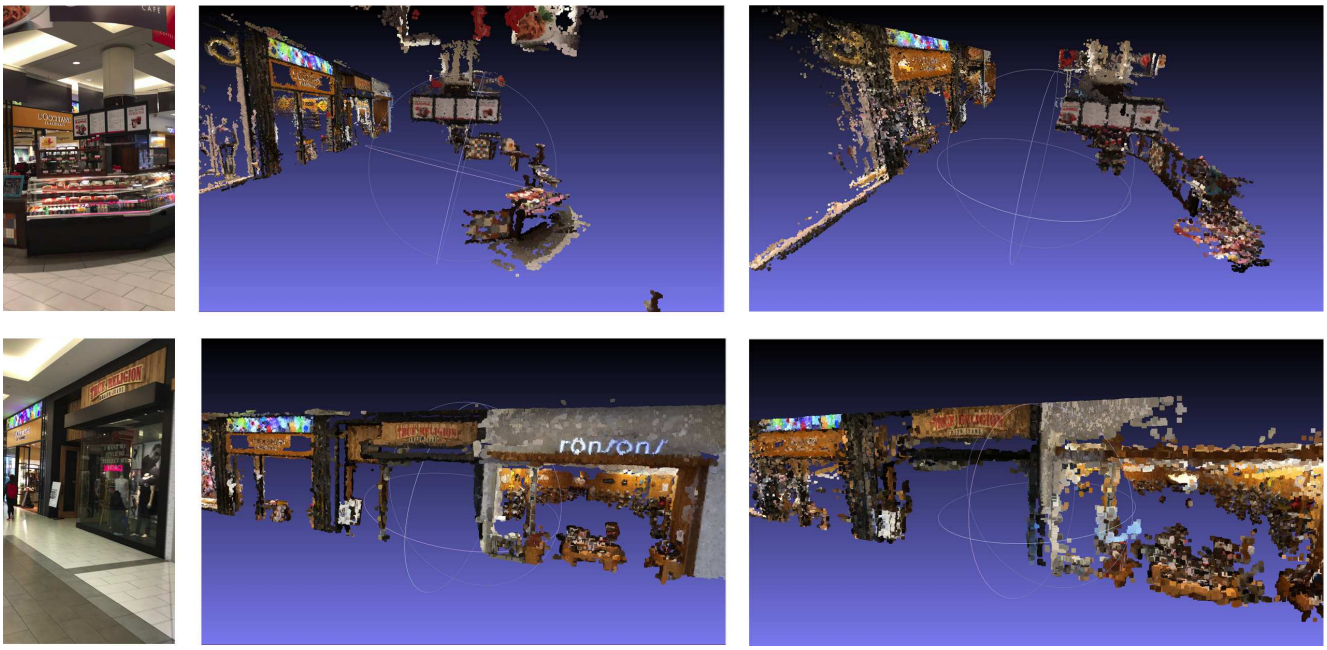


Figure 9: Comparison of 3D visual model details. Left: original video frame; Middle: OpenMVG results; Right: vsfm results.

pose an automation framework that highly utilizes data from 3D reconstruction pipeline to benefit monocular localization pipeline. Through highly automation, we developed a prototype that publishes both virtual navigation and image-based localization services simply using videos as inputs. We demonstrate the performance of our localization accuracy against state-of-the-art method, the comparison between different 3D reconstruction pipelines, and two different web services publishing using our prototype system.

In future work, we aim to pursue further alignment with map data, apply the technology for the whole shopping mall rather than just a section of it, improved localization methods. It is possible that neural network has limitations on the physical area that it can learn and new methods need to push the boundary of recognition accuracy.

References

[1] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski. Building rome in a day. *Commun. ACM*, 54(10):105–112, Oct. 2011.

[2] Aislelabs. `https://www.aislelabs.com/`.

[3] M. A. Brubaker, A. Geiger, and R. Urtasun. Lost! leveraging the crowd for probabilistic visual self-localization. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2013)*, pages 3057–3064, Portland, OR, June 2013. IEEE.

[4] N. Chang, R. Rashidzadeh, , and M. Ahmadi. Robust indoor positioning using differential wi-fi acess points. *Consummer Electronics*, 56(3):1860–1867, July 2010.

[5] H. Du, P. Henry, X. Ren, M. Cheng, D. B. Goldman, S. M. Seitz, and D. Fox. Interactive 3d modeling of indoor environments with a consumer depth camera. In *Proceedings of the 13th International Conference on Ubiquitous Computing*, UbiComp '11, pages 75–84, New York, NY, USA, 2011. ACM.

[6] J. Engel and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *In ECCV*, 2014.

[7] G. Floros, B. van der Zander, and B. Leibe. Openstreetslam: Global vehicle localization using openstreetmaps. In *ICRA*, pages 1054–1059. IEEE, 2013.

[8] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[9] Y. Furukawa, B. Curless, S. M. Seitz, R. Szeliski, and G. Inc. R.: Towards internet-scale multiview stereo. In *In: Proceedings of IEEE CVPR*, 2010.

[10] Indoors. `https://indoo.rs/`.

[11] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, pages 2938–2946. IEEE Computer Society, 2015.

[12] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004.

[13] P. Moulon, P. Monasse, R. Marlet, and Others. Openmvg. an open multiple view geometry library. `https://github.com/openMVG/openMVG`.

[14] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *Int. J. Comput. Vision*, 80(2):189–210, Nov. 2008.

[15] N. Sünderhauf, F. Dayoub, S. Shirazi, B. Upcroft, and M. Milford. On the performance of convnet features for place recognition. *CoRR*, abs/1501.04158, 2015.

[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9. IEEE Computer Society, 2015.

[17] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826. IEEE Computer Society, 2016.

[18] S. Wang, S. Fidler, and R. Urtasun. Lost shopping! monocular localization in large indoor spaces. In *ICCV*, pages 2695–2703. IEEE Computer Society, 2015.

[19] O. Woodman and R. Harle. Pedestrian localisation for indoor environments. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, UbiComp '08, pages 114–123, New York, NY, USA, 2008. ACM.

[20] C. Wu. Towards linear-time incremental structure from motion. In *3DV*, pages 127–134. IEEE Computer Society, 2013.