CONTINUOUS BIM ALIGNMENT FOR MIXED REALITY VISUALISATION

M. Radanovic[1,2,*], K. Khoshelham[1,2], C. S. Fraser[2], D. Acharya[3]

[1] Building 4.0 CRC, Caulfield East, Victoria 3145, Australia
[2] Department of Infrastructure Engineering, The University of Melbourne, Parkville, Victoria 3010, Australia
[3]Geospatial Science, RMIT University, Melbourne, Victoria 3000, Australia
(m.radanovic, k.khoshelham, c.fraser)@unimelb.edu.au, debaditya.acharya@rmit.edu.au

KEY WORDS: Mixed Reality (MR), Deep Learning, Synthetic-real image matching, 3D Model-Based Visual Tracking, Relative camera pose regression.

ABSTRACT:

Several methods exist that can be used to perform initial alignment of Building information models (BIMs) to the real building for Mixed Reality (MR) applications, such as marker-based or markerless visual methods, but this alignment is susceptible to drift over time. The existing model-based methods that can be used to maintain this alignment have multiple limitations, such as the use of iterative processes and poor performance in environments with either too many or not enough lines. To address these issues, we propose an end-to-end trainable Convolutional Neural Network (CNN) that takes a real and synthetic BIM image pair as input to regress the 6 DoF relative camera pose difference between them directly. By correcting the relative pose error we are able to considerably improve the alignment of the BIM to the real building. Furthermore, the results of our experiments demonstrate good performance in a challenging environment and high resilience to domain shift between synthetic and real images. A high localisation accuracy of approximately 7.0 cm and 0.9° is achieved which indicates the method can be used to reduce the camera tracking drift for MR applications.

1. INTRODUCTION

Building Information Models (BIMs) are increasingly being used for the maintenance and operation of buildings where they serve as a digital representation of a building and a database of all related information (Dixit et al., 2019). For the scope of this study, use the term BIM for all parametric building models (BIMs, Digital Twins, CAD models) because we are mainly dealing with the geometry of these models.

Visualisation and interaction with the BIM are conventionally done through a computer screen via mouse and keyboard or a touchscreen, which limits the interpretability and real-time use cases of BIMs. These issues can be addressed by visualising the BIM in 3D and superimposing it on top of the real building using Mixed Reality (MR) visualisation. This truly integrates the model with its underlying source and improves BIM interpretability by providing inherent correspondence between virtual and real building elements (Radanovic et al., 2022). Furthermore, accurate overlay of building models in MR opens up completely new use cases, such as visualisation of objects occluded within the structure or not yet built objects (facility management and planning (Baek et al., 2019), progress monitoring (Kopsida and Brilakis, 2020)), visualisation of routes (navigation and emergency management (Zhu and Li, 2021)) or simulation of specific scenarios (education (Tzima et al., 2019)). Note that we use the term MR, which blends the virtual content with the real world and enables interaction and occlusion between them, as opposed to Augmented Reality (AR), which merely visualises the virtual content on top of the real world (Muthalif et al., 2022).

However, accurate alignment of a BIM to the real building in MR is not a trivial task, especially indoors, where external sensors such as Global Navigation Satellite Systems (GNSS) are not available. The alignment is achieved by performing two tasks, initial alignment and continuous alignment. The initial alignment can be achieved by absolute localisation of the camera within the model, a challenge that has recently seen significant improvements with an onset of deep learning-based pose regression approaches (Acharya et al., 2019a), but also can also be performed by creating simple spatial anchors with fiduciary markers or markerless methods (Marchand et al., 2016). These methods can be used to perform the initial alignment of the BIM to the current camera view, but do not provide continuous alignment.

The second task is maintaining the initial alignment, i.e. performing continuous alignment, which is the focus of the paper. Although continuous estimation of the camera pose in MR, i.e. tracking, can be performed very accurately by different Simultaneous Localisation And Tracking (SLAM) methods, these are local methods that do not maintain the alignment between the BIM and the real building. They estimate the pose of the camera within a sparse 3D model of the environment built during the traversal (Middelberg et al., 2014) and, because there is no relationship between this model and the BIM, the camera will drift the further the device moves from the place of initialisation. The most common way of continuous BIM alignment is model-based camera tracking, which is usually performed by projecting BIM edges based on the initial pose estimate, detecting edges in the real image, and then minimising the distances between them using sampled points (Acharya et al., 2019b). However, this is an iterative approach that requires multiple renderings to solve one frame, and furthermore, these methods are not robust in environments rich with lines due to ambiguities between the edges (Marchand et al., 2016), as well as in frames with a low number of edges, such as close to a wall surface.

The main aim of this paper is to address these issues by pro-

* Corresponding author.

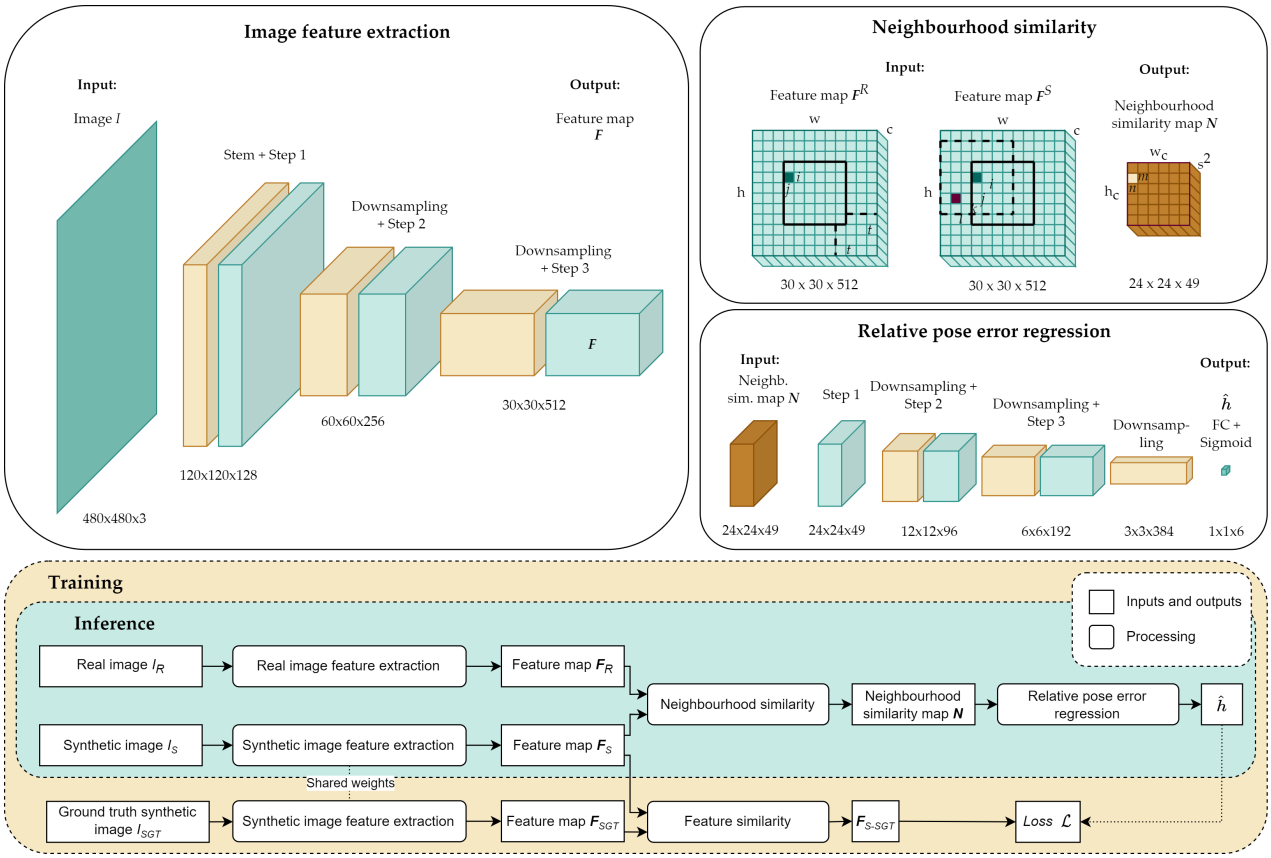

Figure 1. The architecture of the proposed network. The bottom part contains the overall framework. During inference, the network takes as input a pair of images (I_R and I_S) and outputs the 6 DoF relative camera pose error $\hat{\boldsymbol{h}}$ between them. During training, the network takes triplets of images as input and jointly learns to predict errors between transformed pairs (I_R and I_S) and to minimise the difference between extracted feature maps for ground truth pairs (I_R and I_{SGT}), with real encoder weights frozen. The top blocks show the individual processing steps and the output sizes of each step. The feature extraction Steps 1, 2 and 3 are comprised of 3, 3 and 27 ConvNeXt (Liu et al., 2022) blocks, respectively. The relative pose regression Steps 1, 2 and 3 are comprised of 3, 3 and 9 ConvNeXt blocks. The stem layer is a 4-step 4x4 convolutional layer and the downsampling layers are 2-step 2x2 convolutional layers.

posing an end-to-end trainable Convolutional Neural Network (CNN) that takes as input a MR system frame, i.e. a real image and a synthetic image pair, and regresses directly the 6 DoF relative pose difference between them. This pose difference represents a misalignment between the cameras and can be used to correct the misalignment, thus better aligning the BIM to the real building. To better bridge the domain gap between real and synthetic images, we propose a triplet loss function that enforces the description of similar features for the same parts of real and synthetic images, on top of predicting the relative pose error. We evaluate the performance of the proposed network in terms of accuracy and the ability to bridge a domain gap by using a challenging dataset with a low level-of-detail BIM.

Our approach is inspired by an identified knowledge gap, as despite the successes of deep CNNs in 2D image matching and bridging domain gaps, to the best of our knowledge there have been no attempts to apply CNNs for MR camera tracking by relative pose regression.

2. RELATED WORK

The proposed method aims to perform continuous alignment of the BIM to the real building and assumes initial alignment with acceptable accuracy, with position and orientation of the camera determined within 15 cm and 3°. Several methods exist that can achieve initial BIM alignment with acceptable accuracy, and amongst them, the most commonly used are image-based localisation approaches (Marchand et al., 2016). Approaches such as those of (Sarlin et al., 2021; Li et al., 2020; Baek et al., 2019) rely on extracting 2D image features, finding the corresponding matches associated with 3D points within a database, and then using these 2D-3D matches to estimate the camera pose with perspective from points methods (PnP) inside a random sample consensus (RANSAC) loop. An initial alignment can also be made with localisation based on a 3D–3D model registration (Radanovic et al., 2023), which uses a depth camera to register to the existing model of the environment and achieves a mean translation error of 2.8 cm and a mean rotation error of 0.30°. Alternatively, simple fiducial markers can also be used (Marchand et al., 2016) for the initial registration.

The most common use of CNNs for localisation is in CNN-based absolute camera pose regression methods, which have become popular in recent years (Sattler et al., 2019) and which can directly regress an absolute 6 DoF camera pose from a single image. The most popular network that performs the 6 DoF pose regression is PoseNet (Kendall et al., 2015). Many subsequent approaches have been proposed that improve PoseNet's accuracy, such as (Kendall and Cipolla, 2017) which introduces geometric constraints, or (Acharya et al., 2022) which trains on BIM and other synthetic images and performs domain adapta-

Figure 2. Keypoints and reprojection errors before and after correction using the proposed approach. Four top rows show untextured examples and four bottom rows show textured examples. One pair shown per location *Li* (locations shown in Fig. 3). From left to right: real image, the synthetic image before correction, its overlay on the real image and the reprojection errors before correction, synthetic image after correction, its overlay on the real image and the reprojection errors after correction.

tion from BIM to real images and eliminates the need to collect annotated real images. However, absolute pose regression methods essentially only approximate the pose in an approach similar to image retrieval, so they do not achieve accuracy comparable to 3D structure-based methods, and generally cannot generalise beyond training data (Sattler et al., 2019). In contrast, we aim to recognise patterns produced by the misalignment between the two cameras and regress the relative pose difference, which is scene-agnostic.

CNN-based relative pose regression is a more general challenge that may generalise beyond the training data (Yang et al., 2020) and which can be used for continuous alignment of the BIM to the real building. For example, Valada et al. (2018) propose a multitask CNN that jointly performs absolute pose regression and visual odometry estimation from consecutive monocular images, trained with a loss function that enforces the geometrical consistency between the predictions. Furthermore, Yang et al. (2022) propose a visual CNN-based Simultaneous Localisation and Mapping (SLAM) approach which finds the relative camera pose differences between cameras by matching their image features, while Yang et al. (2020) presents a method for estimating the relative camera pose of Unmanned Aerial Vehicles (UAVs). However, these methods assume (consecutive) images from the same sensor, while in contrast, our method must deal with the associated domain shift between real and synthetic images.

The proposed network is mainly inspired by the geometric image matching network introduced by Rocco et al. (2019). They

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-X-1-W1-2023-279-2023 | © Author(s) 2023. CC BY 4.0 License.

propose an end-to-end trainable CNN for matching two 2D images using a geometric model, such as a planar affine transformation. The network is comprised of a Siamese architecture-based feature extractor, which extracts features from both input images, a neighbourhood similarity layer that creates tentative correspondences by computing all pairwise similarities between the extracted feature vectors, and a regression network that robustly estimates the parameters of the geometric model. An improvement is proposed by Rocco et al. (2018a), which aims to handle a large number of computed tentative correspondences by including a soft-inlier count layer that is inspired by RANSAC, which sums the matching scores within a certain distance threshold after the regressed transformation is applied. Another modification (Rocco et al., 2018b) proposes to use 4D convolutions in the 4D space of feature matches to find neighbourhood consensus patterns and propagate information from strong feature matches to uncertain matches, which takes the input of all possible correspondences and outputs filtered correspondences. However, these methods require the geometric transformation between images to be differentiable, and they are weakly or self-supervised and can create pairs or evaluate predictions by applying transformations to images directly, which is not possible for estimating 6 DoF relative pose difference between cameras and creating synthetic images. Furthermore, these methods assume strong and unique descriptor matches between features that can be anywhere in the two images, which cannot be assumed for indoor environments and BIMs that are often uniform with few distinct features.

3. METHOD

As shown in Fig. 1, the proposed network architecture consists of three main parts. During inference, we first employ a pseudo-Siamese architecture with two CNN encoders that extract two sets of feature maps $\boldsymbol{F}_R$ and $\boldsymbol{F}_S$ from input images I_R and I_S. They have the same architecture but different sets of weights. Second, the non-trainable matching layer takes feature maps $\boldsymbol{F}_R$ and $\boldsymbol{F}_S$ as inputs and outputs a neighbourhood similarity map N. Finally, the regressor takes the neighbourhood similarity map as input and regresses to the 6 DoF relative camera pose difference $\boldsymbol{h}$ between the real and the synthetic camera as the final output of the network.

We use an architecture similar to Rocco et al. (2019), but make several key modifications to adapt the network to the task of aligning building models to the real building, as well as several enhancements based on the latest advances in CNN architecture (Liu et al., 2022).

First, to deal with the domain shift in our task, we use a pseudo-Siamese architecture which allows us to train the feature extractors to produce similar feature vectors for real and synthetic images. Next, because indoor environments and especially indoor building models are often uniform and lack distinct features, we modify the neighbourhood similarity layer. Originally, the correlation layer proposed by (Rocco et al., 2019) calculates all pairwise similarities between feature vectors of input images to produce a correlation map, so each feature vector in image 1 is matched to all feature vectors in image 2. In our case, this matching is unnecessary as we perform frame-by-frame tracking and can assume no large viewpoint differences between the frames. Moreover, this strategy does not perform well, as in uniform indoor environments we have an abundance of ambiguous matches and few distinctive features, as can be seen in Fig. 2. Hence, we constrain the pairwise similarities to the immediate neighbourhood to produce a neighbourhood similarity map, explained in detail in the next subsection, which simplifies the search space for the regressor.

We use the recent ConvNeXt-B (Liu et al., 2022) as the backbone feature extractor. Furthermore, inspired by ConvNeXt, which conveniently unites the latest advances in CNN architecture design and training, we construct a new regressor to regress from the neighbourhood similarities map to the 6 DoF error. Note that (Rocco et al., 2019) uses VGG-16 as the feature extractor and a simple succession of two convolutions, which roughly halve the size of the feature map and the number of features, followed by a fully connected layer for the regressor.

3.1 Network architecture

The feature extractor $\mathcal{F}$ is a mapping from the input image $\boldsymbol{I}$ to a 3D tensor $\boldsymbol{F}$, $\boldsymbol{F} = \mathcal{F}(\boldsymbol{I}), \boldsymbol{F} \in \mathbb{R}^{h \times w \times c}$, where h and w are the spatial resolution and c is the number of channels (Dusmanu et al., 2019). Each column feature $\boldsymbol{d}$ of size $1 \times 1 \times c$ can be interpreted as a description of a specific patch of the input image:

$$\boldsymbol{d}_{ij} = \boldsymbol{F}_{ij:}, \boldsymbol{d} \in \mathbb{R}^c \tag{1}$$

where $i = 1, ..., h$ and $j = 1, ..., w$. Tensor $\boldsymbol{F}$, also called a feature or an activation map, can then be interpreted as a dense set of descriptors $\boldsymbol{d}$ describing the image $\boldsymbol{I}$ (Dusmanu et al., 2019).

As shown in Fig. 1, we remove the network head, the final downsampling and the final ConvNeXt step (everything after step 3) from the base network (Liu et al., 2022) in the feature extractor. Unlike most traditional image processing CNNs, ConvNeXt separates spatial and depthwise convolutions and is comprised of spatial downsampling layers (brown in Fig. 1) followed by ConvNeXt blocks (blue).

Feature extractors $\mathcal{F}$ take input images of size $480 \times 480 \times 3$ and output feature maps $\boldsymbol{F}$ of size $30 \times 30 \times 512$. Given two feature maps $\boldsymbol{F}_R, \boldsymbol{F}_S \in \mathbb{R}^{h \times w \times c}$ the matching layer outputs the neighbourhood similarity map $\boldsymbol{N} \in \mathbb{R}^{h_c \times w_c \times s^2}$. We use an approach similar to (Rocco et al., 2019, 2018a,b), but modify the spatial extents of pairwise similarities so that for each descriptor $\boldsymbol{d}^R$ of the real image, only the pairwise similarities of descriptors $\boldsymbol{d}^S$ within the nearest spatial neighbourhood of size s are calculated. We do not use the first $t = \lfloor s/2 \rfloor$ descriptors along the edge of the real image to avoid padding. As all descriptors of the synthetic image are used and all pairwise similarities within the neighbourhood are calculated, this results in minimal loss of information about the misalignment between the images.

The neighbourhood similarity map $\boldsymbol{N}$ is a tensor of size $h_c \times w_c \times s^2$ comprised of cosine similarities:

$$n_{mno} = \langle \boldsymbol{d}^R_{ij:}, \boldsymbol{d}^S_{kl:} \rangle, \tag{2}$$

where $i = t+1, ..., h-t$; $j = t+1, ..., w-t$;
$k = i-t, ..., i+t$; $l = j-t, ..., j+t$;
$m = 1, ..., h-t$; $n = 1, ..., w-t$;
$o = 1, ..., s^2$.

By default, the neighbourhood size is set at $s = 7$, which defines a 7×7 patch of descriptors $\boldsymbol{d}^S$ around each descriptor $\boldsymbol{d}^R$, resulting $o = 49$ similarities in total for each patch. Unlike Rocco et al. (2019) and their later works, we do not normalise the correlation map along the o dimension to penalise ambiguous matches because, in uniform indoor environments, we can expect less distinctive features and many ambiguous patches. Whereas their work aims to find unique matches between input images, we aim to instead perform pattern recognition, a task where CNNs are known to excel (Abiodun et al., 2019), by using the distinctive patterns in the tensor $\boldsymbol{N}$ produced by 6 DoF camera misalignments and large amounts of easily produced synthetic training data.

The final part of the proposed architecture is the regressor, which takes the neighbour correlation map $\mathcal{N}$ and outputs the 6-DoF relative camera misalignment between the two input images. We build a custom regressor based on ConvNeXt (Liu et al., 2022) blocks, with four steps comprised of 3, 3, and 9 blocks followed by a fully connected layer. As the input correlation map is spatially small, we start with depthwise convolutions and perform spatial convolutions, which cause spatial downsampling, from step 2 onward. By halving the output spatial size of each layer while roughly doubling the number of channels the network can learn to recognise increasingly complex patterns of misalignment between input images.

3.2 Loss function and network training

The network is trained in a fully supervised manner with real and synthetic data. As shown in Fig. 1, triplets of images are used for training, containing a ground truth pair I_R and I_{SGT} and a transformed pair I_R and I_S. We generate triplets by taking a real image I_R, generating a synthetic image I_{SGT} at its ground truth pose, then applying a random 6 DoF transformation $\boldsymbol{h}$ and generating a synthetic image I_S. Images I_R and I_{SGT} form a ground truth pair, and images I_R and I_S form a transformed pair. A virtually unlimited number of triplets can be generated for each real image using this strategy.

The network is trained by minimising a triplet loss function, which has two terms. The first term minimises the error between the predicted 6 DoF and the ground truth for the transformed pair, while the second term minimises the difference between the extracted descriptors for the ground truth pair. Forcing similar descriptors in the ground truth pair translates to stronger cosine similarities in the neighbourhood similarity map which helps to resolve the big domain shift between real and synthetic images.

As the network aims to perform frame-to-frame alignment, we can assume the relative translations and rotations between the cameras to be small, and choose a suitable maximum expected values t_{max} and r_{max}. This allows two things. First, we can represent 3D relative rotations with Euler angles for simplicity, as we do not encounter the problem of gimbal lock due to small relative rotations. Second, there is no need for a scaling parameter to keep the values of translation and rotation errors approximately equal, which is widely used in absolute pose regression networks, e.g. (Kendall and Cipolla, 2017). Instead, we can define a normalised 6 DoF relative error $\boldsymbol{h}'$ that our network aims to predict, i.e. the label, as:

$$\underset{6\times1}{\boldsymbol{h}'} = [\ \underset{3\times1}{\boldsymbol{t}'}\quad \underset{3\times1}{\boldsymbol{r}'}\ ]^T = [\ \underset{3\times1}{\boldsymbol{t}}\ /t_{max}\quad \underset{3\times1}{\boldsymbol{r}}\ /r_{max}]^T, \tag{3}$$

where $\boldsymbol{t}$ is the 3D translation and $\boldsymbol{r}$ is the 3D rotation (represented by Euler angles), and each component of the error $\boldsymbol{h}'$ is scaled to the range of $[-1, 1]$. Correspondingly, we add a sigmoid function as the last layer of the regressor to ensure that each component of the predicted 6 DoF relative error, denoted $\hat{\boldsymbol{h}}'$, has values in the range of $[-1, 1]$.

We can now define the triplet loss function as:

$$\mathcal{L} = \|\boldsymbol{h}' - \hat{\boldsymbol{h}}'\|_2 + \beta\|\boldsymbol{F}_R - \boldsymbol{F}_{SGT}\|_2, \tag{4}$$

where the second term ensures the network extracts similar features for the same parts of the positive pair. As a MR visualisation is a composite of two images, a composite of a real and a ground truth synthetic image results in a perfect alignment of the BIM to the real building (in the absence of all errors). If we use the L2 norm as a similarity metric between the extracted feature vectors $\boldsymbol{f}_R$ and $\boldsymbol{f}_{SGT}$ of a ground truth pair, minimising the L2 norm will reduce the difference between them which is expected to aid with the domain shift. Finally, because the first and the second term can be of a significantly different scale, we scale the second term to the magnitude of the first. This ensures both terms, i.e. both goals, are considered equally important and avoids an introduction of a scaling hyperparameter.

3.3 Evaluation metrics

A MR system uses two cameras, a real and a synthetic camera which renders the virtual scene, here the BIM. Assuming a Euclidean space and a pinhole camera, the projection of a 3D world point X to the real camera using the camera matrix C_R is:

$$\underset{3\times1}{\boldsymbol{x}_R} = k_R \underset{3\times4}{\boldsymbol{C}_R}\ \underset{4\times1}{\boldsymbol{X}}\,, \tag{5}$$

where $\boldsymbol{X}$ and $\boldsymbol{x}_R$ are represented in homogeneous coordinates and k_R is a scale factor.

For the scope of this research, we assume the BIM is georeferenced and in the world coordinate frame, and represents the geometry of the real scene correctly. The projection of the same 3D point $\boldsymbol{X}$ to the synthetic camera using the camera matrix $\boldsymbol{C}_S$ is:

$$\underset{3\times1}{\boldsymbol{x}_S} = k_S \underset{3\times4}{\boldsymbol{C}_S}\ \underset{4\times1}{\boldsymbol{X}}\,, \tag{6}$$

where k_S is a scale factor. If there is a misalignment between the projected BIM and the real building, that means the real and the synthetic camera do not have the same pose. Then, the rigid transformation from the synthetic to the real camera can be expressed as:

$$\underset{3\times4}{\boldsymbol{C}_R} = \underset{3\times4}{\boldsymbol{C}_S}\underset{4\times4}{\boldsymbol{H}_S^R}, \tag{7}$$

where $\boldsymbol{H}_S^R$ contains the 3D rotation and 3D translation. In other words, transformation $\boldsymbol{H}_S^R$ is a matrix representation of the 6 DoF relative error $\boldsymbol{h}$ we aim to predict, which aligns the BIM to the real building. We can denote the transformation estimated by our network as $\hat{\boldsymbol{H}}_S^R$, which yields a corrected pose of the synthetic camera $\boldsymbol{C}_{SC}$:

$$\underset{3\times4}{\boldsymbol{C}_{SC}} = \underset{3\times4}{\boldsymbol{C}_S}\underset{4\times4}{\hat{\boldsymbol{H}}_S^R}. \tag{8}$$

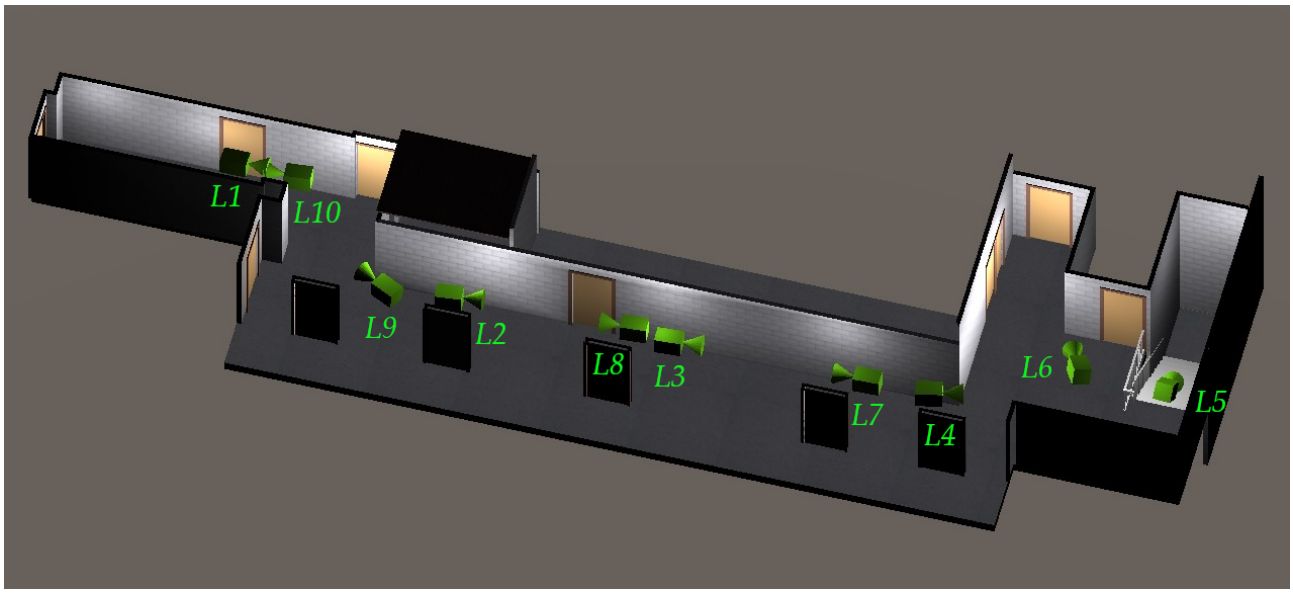

Figure 3. The textured BIM of the testing site, 3rd floor of Infrastructure Engineering Block B, with poses of 10 real cameras indicated. At each location *Li*, 100 synthetic images are created for a total of 1000 pairs used in the experiments.

One way to evaluate this prediction is by comparing it to the ground truth. If we represent the estimated transformation $\hat{\boldsymbol{H}}_S^R$ with a 3D translation vector $\hat{\boldsymbol{t}}$ and a 3D rotation $\hat{\boldsymbol{r}}$ represented by 3 Euler angles, we can calculate the translation and rotation errors as:

$$\begin{aligned} e_t &= \| \underset{3\times1}{\boldsymbol{t}} - \underset{3\times1}{\hat{\boldsymbol{t}}} \|_2, \\ e_r &= \| \underset{3\times1}{\boldsymbol{r}} - \underset{3\times1}{\hat{\boldsymbol{r}}} \|_2. \end{aligned} \tag{9}$$

However, the translation and rotation errors require true camera poses, which may be unknown or unreliable. A way to evaluate our estimation that is independent of the true pose is to measure the reprojection error, which is the average distance between the keypoint pairs in the real and the synthetic image:

$$e_p^S = \sum_{i=1}^{n} \frac{\|x_{Ri} - x_{Si}\|_2}{n}, \tag{10}$$

where $\boldsymbol{x}_{Ri}$ and $\boldsymbol{x}_{Si}$ are projected keypoints of the same 3D point X_i as per (5) and (6), and n is the total number of keypoint pairs.

The process is analogous for the corrected synthetic camera $\boldsymbol{C}_{SC}$ and yields the reprojection error after correction e_p^{SC}. Now, we can compare the reprojection error before the correction, e_p^S, and after the correction, e_p^{SC}, to independently evaluate if the method succeeded in its task, which is to better align the BIM to the real building.

4. EXPERIMENTS AND RESULTS

To evaluate the proposed framework we implement the network in Pytorch and train it on the data gathered within the Block B building of the Department of Infrastructure Engineering at the University of Melbourne. Inference time for one pair is 0.09 seconds on Intel Xeon Gold 6254 3.1 GHZ CPU with an Nvidia A40-16Q 16GB RAM GPU.

4.1 Experimental setup and data preparation

For network training, we use 147 real images gathered by the Microsoft HoloLens (2nd generation) within the Block B building. To retrieve the ground truth poses of the real images, the environment is mapped with the device, which uses an RGB-D SLAM method based on four tracking cameras and a time-of-flight camera (Hübner et al., 2020). The camera poses within this model at the time of capturing the image is taken as the ground truth. For synthetic images, we use a BIM of the third floor of the building which covers an area of approximately 230 m^2 (Acharya et al., 2019a) mainly consisting of a corridor. The model, which is shown in Fig. 3, has a level of development of 300 (Acharya et al., 2019a; Volk et al., 2014) and has been used in several previous works, e.g. (Zhao et al., 2022). We align this BIM to the model of the environment captured by the device, with a combination of manual alignment and ICP, with a Root Mean Square Error (RMSE) of 1.0 cm for a theoretical overlap of 30%. We generate 100 triplets for each real image, resulting in 14700 images for training. The synthetic transformed images are generated by applying a random relative transformation to the synthetic camera, with an experimentally determined t_{max} of 15 cm and r_{max} of 3°.

We evaluate the network on 1000 image pairs in terms of localisation accuracy of the predicted 6 DoF relative camera errors and the alignment accuracy of the BIM to the real world before and after correcting the camera pose, per section 3.3. The 1000 image pairs are generated from 10 real images shown in Fig. 3. These images and pairs have not been used for training the network. Furthermore, to evaluate the effect of the presence of textures on the rendered synthetic BIM images, we create a textured and untextured set of synthetic images (examples shown in Fig. 2) and repeat training and evaluation with both sets.

4.2 Results and analysis

We calculate the translation and rotation error according to Eq. (9) for 1000 pairs. Average errors per location are given in Table 1. The first thing to note is that there are no significant differences between the results from textured and untextured syn-

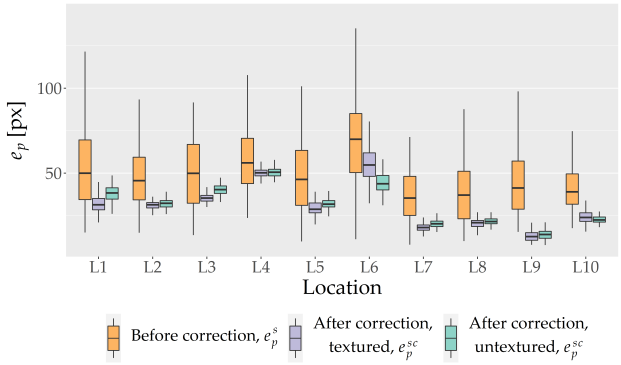

Figure 4. Reprojection errors before and after the correction per location.

thetic images, which indicates the method deals well with the domain shift.

The overall translation error e_t for the textured dataset is 7.05 cm and the overall rotation error e_r is 0.88°. Apart from location L6, the results are fairly consistent across all locations. The results on location L6 may be explained by the differences between the synthetic and the real image, as shown in Fig. 2, with the doors being incorrectly represented and several prominent temporary items missing in the BIM. Similar reasoning may explain the minor differences in results between the rest of the locations, where the elements of the BIM do not precisely represent the elements in the actual scene, hence performing somewhat worse (L1 - missing kitchen elements, L4 - missing ceiling beam and unfaithful railing, L5 - missing door and unfaithful railing, L10 - unfaithful door). On the other hand, the highest accuracy is achieved at locations L2, L3, L7 and L8, which have in common that they visualise a full shot of the hallway and contain no parts with major differences between the BIM and the real building.

We calculate the reprojection errors for 1000 pairs per Eq. (10) and visualise the results in Fig. 4, which is also shown in Table 2. In general, the reprojection errors are much improved after the proposed method is applied, with the mean reprojection error dropping from 48.80 px to 30.86 px (36.8% improvement) for textured and 31.55 px (35.3% improvement) for untextured. To better interpret this result, we should note that the best possible alignment yields reprojection errors ranging from 15-35 pixels, depending on location, due to the inaccuracies of the

	Textured		Untextured	
Location	e_t [cm]	e_r [deg]	e_t [cm]	e_r [deg]
L1	6.95	1.04	8.57	1.18
L2	2.68	0.53	3.30	0.61
L3	5.18	0.35	4.76	0.61
L4	7.76	0.69	8.21	0.87
L5	8.02	1.07	7.90	1.11
L6	13.52	1.91	11.57	2.10
L7	5.05	0.45	4.72	0.48
L8	5.37	0.61	5.44	0.67
L9	6.87	1.01	7.68	1.27
L10	9.04	1.10	8.04	1.02
All	7.05	0.88	7.02	0.99

Table 1. The mean translation and rotation error achieved at each location, based on 100 pairs per location.

Location	Before correction	After correction, textured	After correction, untextured
L1	54.53	31.85	37.86
L2	47.52	31.01	31.86
L3	49.49	35.30	40.16
L4	57.80	50.19	50.51
L5	47.02	29.55	31.89
L6	70.34	54.86	44.84
L7	36.98	18.23	20.18
L8	38.02	20.50	21.50
L9	44.47	12.88	14.00
L10	41.85	24.25	22.72
All	48.80	30.86	31.55

Table 2. The mean reprojection error in pixels (image size 960×540) before and after the correction at each location, based on 100 pairs per location.

BIM and the imperfections of the real camera. The network yields a better reprojection error in 824 and 805 out of 1000 pairs for textured and untextured images, respectively.

The results are consistent with the translation and rotation error results in that the differences between textured and untextured synthetic images are negligible. We show several examples of reprojection errors in Fig. 2. By comparing the initial and corrected alignment in the figure, we can observe that the corrected alignment is almost always considerably better than the initial one. However, we can also observe that the corrected alignment is not always perfect. The cause of this may again be the differences between the BIM and the real building, as above. For example, in L1 we can see that the doors are of incorrect size and that they may never perfectly fit the real doors.

5. CONCLUSION

We have presented a novel approach for continuous alignment of BIMs in MR by regressing to a relative camera pose difference between a real and a synthetic BIM image. Based on the results of the experiments, we conclude that the method considerably improves the alignment of the BIM to the real building, but, this is conditioned by the quality of the BIM and the method may not deliver a high accuracy alignment in areas where there is a discrepancy between the geometry of the BIM and the actual scene. Furthermore, we have shown that the method performs similarly well with textured and untextured BIM images which demonstrates strong resilience to domain shift between real and synthetic images. The proposed approach can be used to reduce the camera tracking drift for MR BIM applications in near real-time on the current generation of hardware.

6. ACKNOWLEDGEMENTS

This research is supported by Building 4.0 CRC. The first author acknowledges the financial support from the University of Melbourne through the Melbourne Research Scholarship.

References

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Umar, A. M., Linus, O. U., Arshad, H., Kazaure, A. A., Gana, U.,

Kiru, M. U., 2019. Comprehensive review of artificial neural network applications to pattern recognition. *IEEE Access*, 7, 158820–158846.

Acharya, D., Khoshelham, K., Winter, S., 2019a. BIM-PoseNet: Indoor Camera Localisation Using a 3D Indoor Model and Deep Learning from Synthetic Images. *ISPRS J. Photogramm. Remote Sens.*, 150, 245–258.

Acharya, D., Ramezani, M., Khoshelham, K., Winter, S., 2019b. BIM-Tracker: A Model-Based Visual Tracking Approach for Indoor Localisation Using a 3D Building Model. *ISPRS J. Photogramm. Remote Sens.*, 150, 157–171.

Acharya, D., Tennakoon, R., Muthu, S., Khoshelham, K., Hoseinnezhad, R., Bab-Hadiashar, A., 2022. Single-Image Localisation Using 3D Models: Combining Hierarchical Edge Maps and Semantic Segmentation for Domain Adaptation. *Autom. Constr.*, 136, 104152.

Baek, F., Ha, I., Kim, H., 2019. Augmented Reality System for Facility Management Using Image-Based Indoor Localization. *Autom. Constr.*, 99, 18–26.

Dixit, M., Venkatraj, V., Ostadalimakhmalbaf, M., Pariafsai, F., Lavy, S., 2019. Integration of Facility Management and Building Information Modeling (BIM): A Review of Key Issues and Challenges. *Facilities*, 37(7-8), 455–483.

Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A., Sattler, T., 2019. D2-Net: A Trainable CNN for Joint Description and Detection of Local Features. *Proc. CVPR IEEE*, IEEE, Long Beach, CA, USA, 8084–8093.

Hübner, P., Clintworth, K., Liu, Q., Weinmann, M., Wursthorn, S., 2020. Evaluation of HoloLens Tracking and Depth Sensing for Indoor Mapping Applications. *Sensors*, 20(4), 1021.

Kendall, A., Cipolla, R., 2017. Geometric Loss Functions for Camera Pose Regression with Deep Learning. *Proc. CVPR IEEE*, IEEE, Honolulu, HI, 6555–6564.

Kendall, A., Grimes, M., Cipolla, R., 2015. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. *2015 Proc. ICCV IEEE*, IEEE, Santiago, Chile, 2938–2946.

Kopsida, M., Brilakis, I., 2020. Real-Time Volume-to-Plane Comparison for Mixed Reality–Based Progress Monitoring. *J. Comput. Civ. Eng.*, 34(4), 04020016.

Li, J., Wang, C., Kang, X., Zhao, Q., 2020. Camera Localization for Augmented Reality and Indoor Positioning: A Vision-Based 3D Feature Database Approach. *Int. J. Digit. Earth*, 13(6), 727–741.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., Xie, S., 2022. A convnet for the 2020s. *Proc. CVPR IEEE*, 11976–11986.

Marchand, E., Uchiyama, H., Spindler, F., 2016. Pose Estimation for Augmented Reality: A Hands-On Survey. *IEEE Trans. Vis. Comput. Graph.*, 22(12), 2633–2651.

Middelberg, S., Sattler, T., Untzelmann, O., Kobbelt, L., 2014. Scalable 6-DOF Localization on Mobile Devices. D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (eds), *Computer Vision – ECCV 2014*, 8690, Springer International Publishing, Cham, 268–283.

Muthalif, M. Z. A., Shojaei, D., Khoshelham, K., 2022. A Review of Augmented Reality Visualization Methods for Subsurface Utilities. *Adv. Eng. Inform.*, 51, 101498.

Radanovic, M., Khoshelham, K., Fraser, C., 2022. Virtual Element Retrieval in Mixed Reality. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.*, V-4-2022, 227–234.

Radanovic, M., Khoshelham, K., Fraser, C., 2023. Aligning the Real and the Virtual World: Mixed Reality Localisation Using Learning-Based 3D–3D Model Registration. *Adv. Eng. Inform.*, 56, 101960.

Rocco, I., Arandjelović, R., Sivic, J., 2018a. End-to-end weakly-supervised semantic alignment. *Proc. CVPR IEEE*, 6917–6925.

Rocco, I., Arandjelovic, R., Sivic, J., 2019. Convolutional Neural Network Architecture for Geometric Matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(11), 2553–2567.

Rocco, I., Cimpoi, M., Arandjelović, R., Torii, A., Pajdla, T., Sivic, J., 2018b. Neighbourhood Consensus Networks. *Adv. Neural. Inf. Process. Syst.*, 31.

Sarlin, P.-E., Unagar, A., Larsson, M., Germain, H., Toft, C., Larsson, V., Pollefeys, M., Lepetit, V., Hammarstrand, L., Kahl, F., Sattler, T., 2021. Back to the Feature: Learning Robust Camera Localization from Pixels to Pose. *Proc. CVPR IEEE*, IEEE, Nashville, TN, USA, 3246–3256.

Sattler, T., Zhou, Q., Pollefeys, M., Leal-Taixe, L., 2019. Understanding the Limitations of CNN-Based Absolute Camera Pose Regression. *Proc. CVPR IEEE*, IEEE, Long Beach, CA, USA, 3297–3307.

Tzima, S., Styliaras, G., Bassounas, A., 2019. Augmented Reality Applications in Education: Teachers Point of View. *Education Sciences*, 9(2).

Valada, A., Radwan, N., Burgard, W., 2018. Deep Auxiliary Learning for Visual Localization and Odometry. *Proc. ICRA IEEE*, 6939–6946.

Volk, R., Stengel, J., Schultmann, F., 2014. Building Information Modeling (BIM) for Existing Buildings — Literature Review and Future Needs. *Autom. Constr.*, 38, 109–127.

Yang, C., Chen, Q., Yang, Y., Zhang, J., Wu, M., Mei, K., 2022. SDF-SLAM: A Deep Learning Based Highly Accurate SLAM Using Monocular Camera Aiming at Indoor Map Reconstruction With Semantic and Depth Fusion. *IEEE Access*, 10, 10259–10272.

Yang, C., Liu, Y., Zell, A., 2020. RCPNet: Deep-Learning based Relative Camera Pose Estimation for UAVs. *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, 1085–1092.

Zhao, Y., Zhao, H., Radanovic, M., Khoshelham, K., 2022. A Unified Framework for Automated Registration of Point Clouds, Mesh Surfaces and 3D Models by Using Planar Surfaces. *The Photogrammetric Record*, 19.

Zhu, Y., Li, N., 2021. Virtual and augmented reality technologies for emergency management in the built environments: A state-of-the-art review. *J. Saf. Sci. Resil.*, 2(1), 1–10.