POSE ESTIMATION THROUGH MASK-R CNN AND vSLAM IN LARGE-SCALE OUTDOORS AUGMENTED REALITY

A.-M. Boutsi, N. Bakalos, C. Ioannidis

Laboratory of Photogrammetry, School of Rural and Surveying Engineering, National Technical University of Athens, Greece

Commission IV, WG IV/9

KEY WORDS: Deep Learning, Augmented Reality, CNN, image recognition, pose estimation, 3D rendering

ABSTRACT:

Deep Learning (DL) ingrained into Mobile Augmented Reality (MAR) enables a new information-delivery paradigm. In the context of 6 DoF pose estimation, powerful DL networks could provide a direct solution for AR systems. However, their concurrent operation requires a significant number of computations per frame and yields to both misclassifications and localization errors. In this paper, a hybrid and lightweight solution on 3D tracking of arbitrary geometry for outdoor MAR scenarios is presented. The camera pose information obtained by ARCore SDK and vSLAM algorithm is combined with the semantic and geometric output of a CNN-object detector to validate and improve tracking performance in large-scale and uncontrolled outdoor environments. The methodology involves three main steps: i) training of the Mask-R CNN model to extract the class, bounding box and mask predictions, ii) real-time detection, segmentation and localization of the region of interest (ROI) in camera frames, and iii) computation of 2D-3D correspondences to enhance pose estimation of a 3D overlay. The dataset holds 30 images of the rock of St. Modestos – Modi in Meteora, Greece in which the ROI is an area with characteristic geological features. The comparative evaluation between the prototype system and the original one, as well as with R-CNN and FAST-R CNN detectors demonstrates higher precision accuracy and stable visualization at half a kilometre distance, while tracking time has decreased at 42% during far-field AR session.

1. INTRODUCTION

Deep Learning (DL) has a profound impact on how data are structured, analysed and interpreted. Instead of a linear, sequent and complex logic, multi-layered neural networks achieve one-shot computation and learning. The most commonly used for visual tasks are Convolutional Neural Networks (CNNs). CNNs have simplified object recognition by classifying or assigning labels to image datasets based on their concept. They can output bounding box coordinates predicting objects 2D relative position fast and efficiently enough for real-time mobile applications. Their key benefits over other algorithms in the field are the weight sharing and the automatic detection of features without any human supervision (Alzubaidi et al., 2021). To harness their potential to the fullest, CNNs are coupled with computer vision and Augmented Reality (AR). AR allows for displaying virtual information aligned with their real-world position through a camera-equipped device. Therefore, this fusion approaches the close relationship between human visual perception and memory. The captured visual information is processed and interpreted based on prior experience, and then, translated into insights used to drive decision making.

At its core, AR is a motion tracking issue involving the concepts of features detection and matching, photogrammetric bundle adjustment, homography estimation and 6 Degrees of Freedom (DoF) tracking (Marchand et al., 2016). It entails stable poses and accurate localization across successive image frames derived from the camera feed in real-time. Computer vision methods based solely on features points like Structure from Motion (SfM), often lead to large registration errors while localization with visual simultaneous localization and mapping (vSLAM) (Durrant-Whyte and Bailey, 2006) or Visual Odometry (VO) (Nister et al., 2004) is prone to partial occlusion. On the other hand, outdoors GPS positioning is still inaccurate and subject to shifting by a certain offset and coordinate changes over time. Although DL underlies end-to-end solutions to 6 DoF pose estimation, temporal stability of the tracking is hard to handle by the neural network directly.

The presented work examines the synergy of a lightweight instance segmentation Mask-R CNN model with the 6 DoF tracking of ARCore for complex and large-scale outdoor environments. Besides a class label and a bounding-box offset, Mask-R CNN outputs the object mask. In the proposed methodology, the mask output and its relative screen ratios are guiding the actual feature-point regression for pose estimation with 2D-3D correspondences. The network is successfully trained to distinguish between 5 types of land use and 3 types of different rock material and formations. On AR session, the model recognizes the candidate area, gets its shape and relative screen position while texture masking limits the detector's window size and reduces the computational load of SLAM during features extraction. Finally, the 2D coordinates of the boxes are used against ARCore's Hit test to determine where a 3D asset should be accurately overlaid. To our knowledge, the generalization of MASK-R CNN in the context of AR is briefly addressed in the literature. Moreover, high tracking accuracy and stable visualization are achieved for long-distance overlays while tracking and 3D models' drawing times are decreased up to 42% and 32% accordingly. The attained performance demonstrates that vSLAM is significantly optimized by the adoption of the observational redundancy of the photogrammetric space resection and the progressive refining strategy driven by a simple segmentation CNN model.

Case study is the rock of St. Modestos or Modi, located at the UNESCO site of Meteora, Greece. The region of interest (ROI) is part of the rock summit with adjacent physical and visual geological features that define a characteristic pattern. The 3D

photorealistic reconstruction of the no longer existent monastery of St. Modestos, traces of which are still visible on the pinnacle of the rock, is the 3D asset, superimposed on the center of the ROI. Besides lightweight 3D geometries, the low-cost outdoor mobile AR application can visualize and integrate into their unknown physical environments high-resolution 3D models acquired by close-range or aerial photogrammetry and derived by image-based 3D reconstruction. By integrating a relative training dataset and fine tuning only the upper convolutional layers, the learning can be transferred in related situations. Cultural Heritage sites, natural landscapes, urban spaces and residential areas can be augmented by high-resolution models for the purposes of education, land management, construction, architecture and navigation. The rest of the paper is organized as follows: Section 2 reviews related methods and works, current advancements as well as research challenges. In Section 3, the network architecture and the steps of the proposed methodology are described in detail. Section 4 outlines the implementation and results of CNN training and AR application. Section 5 assesses them in comparison with those ones achieved through other CNN detectors and pure computer vision. Finally, Section 6 concludes the proposed work and depicts the future research steps.

2. RELATED WORK

2.1 Deep Learning

The advances in sensors and hardware of mobile devices have expanded the use of DL to real-life applications. That pace will continue to accelerate, thanks to a large number of DL platforms and frameworks ported to Android operation system, such as TensorFlow Lite (Google Inc., 2021), MXNet (Apache Team, 2021) for on-device inference and deployment as well as libraries, such as ML Kit (Google Inc., 2021), Keras (Keras Team, 2021), and PyTorch (Paszke et al., 2019), that undertake training. All of the above APIs are optimized for mobile integrations but ML Kit is the only one that bundles together Google's machine learning algorithms, TensorFlow Lite and the android network API (Baruah et al, 2021).

Among the various architectures of neural networks (RvNNs), Convolutional Neural Networks (CNNs) are currently serving as the main method of DL in mobile vision tasks and spatial data. This is due to the fact that convolutional layers in CNNs allow for fast and reliable feature extraction from visual data, inherently taking into account spatial correlations. This enables even simple CNN architectures to extract semantic information from visual sources (Bakalos et al., 2020). Region-Based CNN (R-CNN) model is a multi-stage approach of object detection and semantic segmentation comprising of three modules: region proposal, feature extraction and classifier (Girshick et al., 2014). The model family has evolved with the techniques of Fast R-CNN (Girshick., 2015) and Faster-RCNN (Ren et al., 2016). Despite the high level of accuracy, these models are not speed-optimized. Real-time use embraces unified architectures and one-stage pose estimators that detect ROIs containing target objects and predict concurrently their class labels (Martinez-Alpiste et al., 2021). YOLOv4 is the latest and most accurate performant model of YOLO family algorithms for real-time detection on conventional GPUs (Bochkovskiy et al., 2020) while SSD (Single Shot Detector) foresees the bounding boxes and the categorization probability (Liu et al., 2016). It divides their space into several anchor boxes at a variety of aspect ratio compared to Faster-RCNN but both detectors, do not regress 3D bounding boxes directly and are not suited to AR applications. Since AR is an already intensive task, their concurrent operation for the recovery of the 3D poses from a frame sequence captured by the live camera involves many steps of intense computation. Furthermore, humidity conditions, dynamic lighting, varying transformations in size, rotation and aspect ratios as well as occlusion yield to false positive feature detection and inaccurate alignment. Thus, the rest of the literature review emphasizes on image classification and object recognition methods adapted to the limited computing power and energy available of mobile devices. Related works concern the use of DL as a means of optimization of 6 DoF pose estimation during an AR session and not as a direct solution.

2.2 Marker-less Mobile Augmented Reality

A state-of-the-art improvement of Faster R-CNN is Mask-R CNN (He et al., 2017). It adds a branch for predicting segmentation masks on each ROI while simultaneously, it classifies and generates their bounding boxes. Its high-performance precision and inference speed are acceptable in real-time usage, considering the addition of the segmentation process in the architecture. Mask-R CNN outperforms YOLOv3 and YOLOv4 when transformations like rotation and scaling are applied to image datasets (Zhang et al., 2021). Only one study has examined its integration in AR, attaining a marker detection accuracy of approximately 70% (Perdpunya et al., 2021).

The detected objects or areas along with their masks can be post-processed in order to regress object poses in 3D space. The presented work uses their screen coordinates as extra geometric parameters to overlay 3D models on their real-world position accurately and fast. A similar method of pose and shape estimation from 2D bounding boxes uses MultiBin, a deep CNN architecture, for orientation prediction and choice of box dimensions as regression parameters (Mousavian et al., 2017). Our prototype is built with ARCore SDK for Android operating systems which integrates vSLAM for simultaneously locating the camera position and mapping the real world from a monocular camera (Google Inc., 2021). Other SDKs dedicated for MAR adopt either vSLAM, like ARKit (Apple Inc., 2021) for iOS and WikiTube (Qualcomm Inc., 2021) or variations of the algorithms. EasyAR (EasyAR Inc., 2021) and Vuforia (PTC Inc., 2021) fuse camera and IMU data in the context of VISLAM (Visual-Inertial SLAM) method. One key difference lies on the ability of VISLAM to estimate scale accurately and consistently through the inertial measurements (Jinyu et al., 2019). In general, the synergy of the aforementioned AR software with DL architectures is an active research area with a variety of applications in medicine (Pauly et al, 2015), retail industry and advertising (Upadhyay et al, 2020), manufacturing (Sahu et al., 2021), robotics (Bassyouni and Elhajj, 2021) and weather forecast (Freeman, 2020).

The adaptation of DL object detection, registration of virtual overlays and tracking under uncontrolled outdoor conditions has also been addressed Rao et al. (2017). The results of a modified version of SSD network are spatially associated with the sensor data of a GPS receiver, the Inertial Measurement Unit (IMU) and the magnetometer attain precise registration and sufficient robustness during tracking. One year later, Wang et al tackled outdoor localization with a visual-GPS fusion method. After geohash conversion and ORB feature detection, a Faster R-CNN detector matches the recognized scene with the target while RANSAC removes wrong match results and estimates camera's homography (Wang et al., 2018). Finally, DeepMobileAR is a mobile AR application that merges ORB-SLAM with a SSD

object detection framework (Lee et al., 2019). From a performance standpoint for mobile AR, only a few approaches can be considered as complete. Liu et al. (2019) offload CNN object detection to edge cloud computing and calculate only the valid region of the output feature map of the network in order to reduce latency and improve detection accuracy. MediaPipe Objectron developed by Google, exploits the ground truth camera pose information from the AR session data to detect 3D objects from a single RGB image (Ahmadayan et al., 2020). More recently, a YOLOv3 detection model is incorporated to an AR prototype that classifies any images as AR markers, detects and tracks them in real-time (Le et al., 2021). Finally, apart from enhancing the functionality of AR with DL, the inverse problem has also been tackled. The received image frames as well as the interior and exterior orientation parameters of the camera can be employed to generate labelled or synthetic training datasets (Su et al., 2021).

3. METHODOLOGY

3.1 Neural Network Architecture

Region-based CNNs or regions with CNN features (R-CNNs) are among the best performing approaches in applying deep learning paradigms to object detection. These architectures first extract region proposals form the input image. These regions act as ground truth data and CNN forward propagation is used on each region to extract representative features that are then used in the classification process of the region under analysis, producing as a final output a bounding box of this region proposal. The R-CNN approach can be broken down into four main steps. Selective search is initially performed to identify region proposals. For each proposal a usually pretrained CNN is used to extract features. These features and the region proposal label is used to train multiple binary support vector machines (SVM). In this step the number of SVM classifiers is equal to the number of predicted classes. Finally, a linear regression model predicts the ground-truth bounding box.

To compensate for the performance loss during the extraction of salient features from each individual region, a Fast R-CNN approach is used where the CNN extracts features for the entire image, allowing for a single forward propagation to extract features for all the regions. Moreover, a region of interest pooling layer is used. Specifically, a Mask-R CNN approach is applied, where the ROI pooling layer is substituted by a ROI alignment layer and a fully convolutional CNN (FCNN) is used to predict the necessary mask. The overview of the network architecture can be seen in Figure 1.

3.2 Mask-R CNN in Augmented Reality

TensorFlow Lite ports the trained model to Android converting it to a FlatBuffer (tflite) file. The Mask-R CNN instance gets the camera frame as bitmap and outputs a bitmap of the same width and size. The input and output of the model are on the GPU side, and the only CPU calculations concern extracting the bitmap and passing it into the GPU side. The camera frame retrieval, segmentation mask texturing and object detection operate at different threads. The code iterates the number of the detected regions that are ranked with a score associated with the probability of correct prediction. Only the most confident region is kept ensuring that its confidence is greater than the minimum one. Then, a class-independent segmentation is performed, targeting the attention ROI.

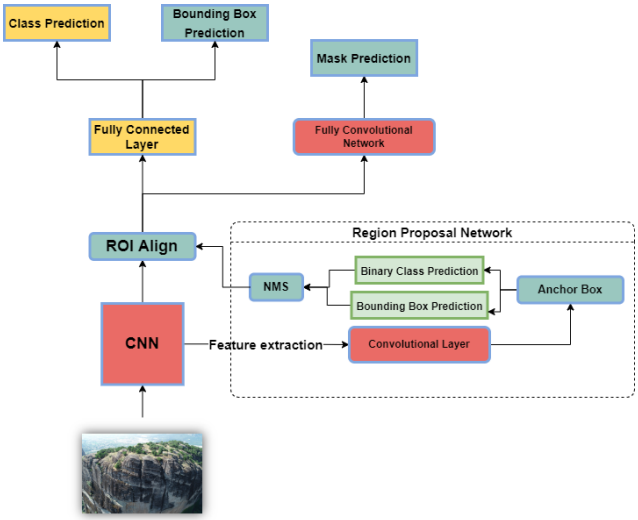

Figure 1: Architecture of the Mask-R CNN

The predicted mask is resized to fit the image dimensions, inverted and assigned to a SurfaceView object of the graphics API. SurfaceView renders transparent content and has increased performance in compositing the masks over camera view. The output bitmaps are sent to SLAM for motion analysis. Concurrently, bounding rectangles are calculated from mask contours and stored into a buffer array. Thus, only the potential ROI participates in Mask-R CNN's detection as well as in the feature matching and bundle adjustment of vSLAM. At the succeeding of object localization, the network performs a bounding box regression and then, adjusts its weights and biases optimizing for bounding box predictions.

3.3 Robust Pose Estimation

The integrated vSLAM algorithm implements feature extraction, feature matching between frames and camera motion estimation including loop detection and loop closure. Since the detection area of the frame is reduced by the mask layer, the potential outliers and noise are minimized. Then, ARCore creates a line extending from the camera and casting into the scene. If ray casting occurs, a list of arrays that exposes the collection of collision points or planes is populated. ARCore uses Anchors, tracked physical locations, to determine the position where the virtual object is to be drawn, in relation with the ray casting results. Since they are subject to drifting effects and misplacement when distance exceeds 10 meters, long-distance registration is estimated by the 2D-3D correspondences.

Once ray casting, the SLAM's keyframe and the initial camera pose matrix are extracted. Whether the previous frame is a key frame, the process of pose refinement is initialized. The following geometric and mathematical techniques are inherited by the resectioning problem and depict the strong relationship between the proposed CNN-based AR and photogrammetry. The inner array of the tensors represents the bounding boxes as four floats in the range [0.0, 1.0], which are converted to actual screen coordinates. Given the pose and assuming that camera interior orientation parameters are known, the origin of the world coordinate system is translated into the projection center, which is the origin of the 3D Cartesian camera system. Concurrently, the world coordinate system is rotated into the camera system. This is achieved by passing the joint rotation-translation matrix [R|t] of camera pose in the Model View Projection (MVP) sequence of OpenGL's transformations. The computation returns

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-V-4-2022-197-2022 | © Author(s) 2022. CC BY 4.0 License.

a float array containing the location followed by a normalized direction vector (Figure 2).

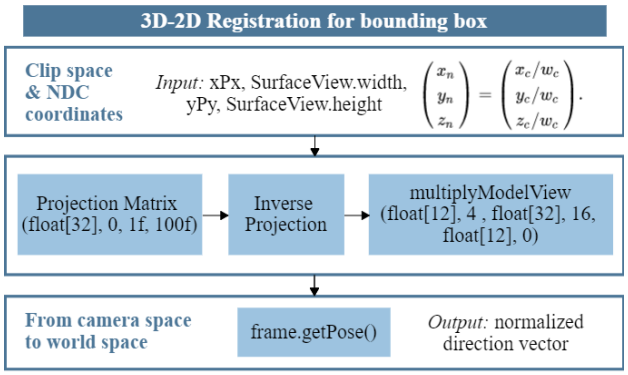

Figure 2: World coordinate frame ray computation for a screen point.

Their 2D-3D correspondence is recovered and the new frame rays are added to the ARCore's list. Pose estimation is recalculated and an Anchor is created based on the generated coordinates, that it is ideally, located at the center of the bounding box. The 3D model is displayed with a rotation perpendicular to the estimated surface normal of the Anchor. By default, its biggest dimension equals to the biggest dimension of the bounding box. The limited field of view on the screen makes it difficult for the end-users to perceive depth, scale and distance. Placing the model within long distances from device's camera makes it appear small. For far-field AR, at distances of around 200 – 500 meters, a scale adjustment is designed. The length of the biggest edge of the bounding box is divided with four time its size in world meters.

4. PROTOTYPE SYSTEM

4.1 Experimental Setup

All the aspects of the Mask-R CNN were trained using a captured dataset. The initiall CNN that is able to export salient features from the input, and also to produce the necessary masks, that will act as the annotation during training.

4.1.1 Input Data: The dataset contains 30 images of 5 types of land use while the pillar consists of 3-4 types of different material. The ROI is located at the summit of the rock and it is marked by vertical lines. These physical morphological formations serve as the characteristic feature points that enhance the ROI detection task. Throughout the imagery, it can be perceived from diverse viewpoints and transformations in size and rotation.
To appropriately train the data, an annotation of the dataset took place. The annotation process is based on two main steps. Firstly, the sections of the image that represent solid rock ground were appropriately identified. Then, the areas on the top of the rock that are appropriate for the 3D model were identified from these sections of the image. The training dataset underwent augmentation and then it was fed to an extended K-means clustering algorithm that grouped areas with visual similarity. To improve the clustering performance, the image input was augmented transforming the 3D multicolour image to a larger 3D-cube. Beyond the information about the different colour channels, information about the edges and the image entropy is also stored. Thus, the initial dataset that included images with a resolution of 5472x3648x3, is transformed in a resolution of 5472x3648x5. These images were fed into a k-means cluster that created a two-dimensional cluster map of using them as input. Suitable clusters were selected to disregards regions that were not part of the desired ROI. This set created the image mask that was used for training. To smooth the output, and also, to remove outliers, a 20x20 median filter was applied to all the generated masks.

4.1.2 Training: From the 30 images available, 6 images were also used to train the CNN of the architecture developed for the feature extraction step. This CNN, has an autoencoder like structure, i.e., the input and output channels are of the same size. Only the encoding part was used, for the feature extraction step of the Mask-R CNN. The rest of the images were split into training and testing sets, containing 19 images and 5 images respectively. To increase the training samples for the feature extraction step, non-overlapping patches of 228x228 resolution were considered, while the resolution of these non-overlapping patches was changed to 556x556. In all the images, flipping, and image transformations under different brightness and saturation values were used to augment the dataset. This resulted in a total of 3072 training samples of a 228x228x5 resolution for the feature extraction and 18432 samples of a 556x556x5 resolution for the Mask-R CNN. All the learning models, including various clustering and other CNN algorithms that were exploited, were implemented and trained in the Google Colab infrastructure, using Tensorflow and Keras learning libraries.

4.1.3 Accuracy Testing: To assess the performance of the Mask-R CNN on the dataset, the Average Precision metric was used over an IoU of 0.5. This metric offers a better understanding of the overall performance of the model, in comparison with simple precision and recall values. While precision indicates the percentage of correct predictions, and recall how well the positives are found, the AP metric represents the area under the precision-recall curve. Moreover, because of the alignment of the ROI and the predicted ROI, a simple pixel to pixel comparison does not suffice. Thus, the overlap of the prediction with the ground-truth is measured with a threshold of 0.5. The selected value means that, if the area of overlap between the prediction and the ground truth is at least half of the area of the union of the two, a classification is considered to be successful. Table 1 presents the results of this analysis. To assess the performance, it was compared with a Fast R-CNN architecture of the same structure but with a ROI pooling instead of a ROI Alignment layer, as well as with a simple R-CNN. In the table, the Accuracy, Precision, Recall and AP^{50} are presented along with the time needed to receive an output from each method.

Table 1. Performance comparison among R-CNN, Fast-R CNN and Mask-R CNN.

Method	Metrics				Time (Sec)
	Accuracy (%)	Precision (%)	Recall (%)	AP^{50} (%)	
R-CNN	81.27	23.56	48.00	48.23	2
Fast-R CNN	79.81	23.28	54.00	52.60	0.2
Mask-R CNN	85.05	34.32	72.00	68.84	0.24

4.2 System architecture of AR prototype

The Keras classifier is converted to a mobile-optimized network architecture, TensorFlow Lite model. The model is bundled with the application and a local interpreter is created to inspect it into a native session. ML Kit Android library performs on-device inference. To invoke model acceleration, inference takes place on the GPU backend side in 32-bit float precision. Despite the lack of native rendering engines, ARCore supports OpenGL ES

rendering. Figure 3 summarizes the development flow and illustrates the methodology described in Sections 3.2 and 3.3.

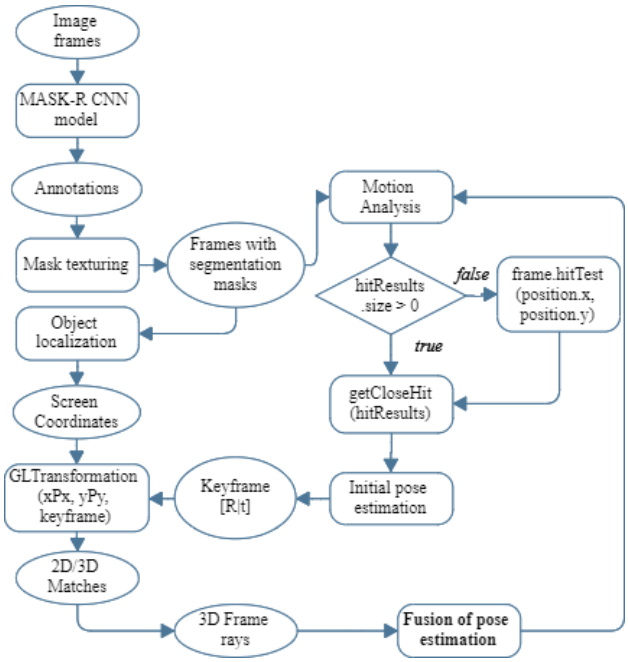

Figure 3: World coordinate frame ray computation for a screen point.

4.3 Case study & Results

The asset for the AR prototype is a photorealistic 3D mesh of wavefront (OBJ) format. The color and texture information are stored in an associated MTL file along with 21 UV texture images. It comprises 36K vertices and 60K faces, corresponding to a size of 6 MB. The mesh is a 3D representation of the destroyed Byzantine monastery of St. Modestos – Modi. Surface traces and ruins of its tower are still visible on the ridge of the rock. These archaeological evidences along with the interpretative hypotheses based on coeval constructions lead to the determination of the geometry, size and spatial position of the no-longer existing structures (Figure 4).

Figure 4. The 3D model of the monastery of St. Modestos.

Once the ROI is detected, the 3D model is overlaid at the given pose and scaled to its real-world proportions (Figure 5). However, the visualization is not consistent down to the details due to a drop in texture quality (Figure 5a). To preserve its realistic appearance, normal maps are used for texturing and Physically Based Rendering (PBR) is enabled (Figure 5b). While there is no visual mismatching in the case of the far-field overlay, the occlusion problem has not been solved, reducing realism consistency. Addressing its size perception from half a kilometre distance, its length is triple the width of its bounding box. It can be distinguished from the surrounding natural landscape and the end-user does not under-estimate its size against the real scene depth (Figure 5c).

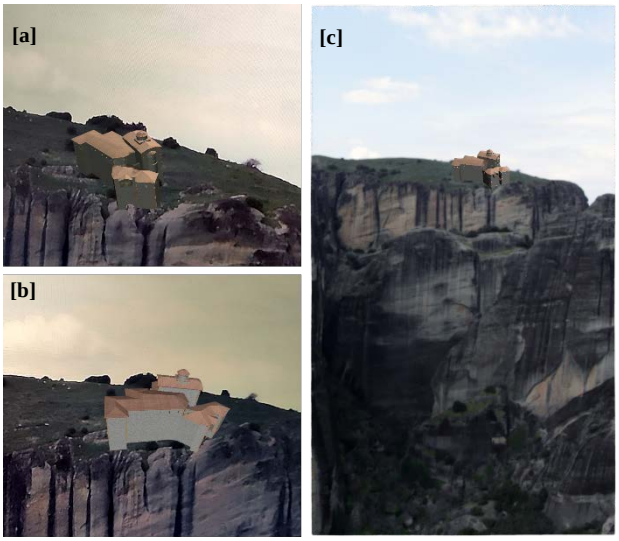

Figure 5. Screenshots of the overlaid 3D reconstruction applying: [a] diffuse shading, [b] Physically Based Rendering and [c] Scale adjustment.

5. EVALUATION

5.1 Mask-R CNN algorithm

The AP^{50} metric showcases that the Mask-R CNN outperforms the other classifiers. The differences in Precision imply significant alignment issues; however, Mask-R CNN manages to compensate for these misalignments. Moreover, high accuracy is achieved by all tested methods. This indicates that R-CNN architectures can at the very least identify sufficiently areas of interest. Finally, the time requirements of a simple R-CNN network are too high to be used in the proposed application. This was expected, as the need to do multiple forward propagations in the CNN for each region label creates a computational overhead. However, both Fast-R CNN and Mask-R CNN approaches demonstrate sufficient time complexity, that will allow the prediction of the ROI multiple times per second.

5.2 Registration and Tracking

The evaluation tests are executed on a high-end smartphone with a Snapdragon 720G CPU at 2.30 GHz, 8 GB of RAM, a resolution of 1080x2400 pixels and 60Hz refresh rate. First of all, the registration accuracy is analyzed using the root mean square error (RMSE), of the distances between the inliers and their reprojected 3D reference features using the refined estimation of the camera pose, as expressed in equitation (1).

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(A_i - P_i)^2} \qquad (1)$$

where N = number of samples
A_i = detected points in the keyframe
P_i = observed points

Table 2 summarizes the average RMSE values as ratios of correct distances (in pixels) between ground truth pairs of corresponding points to their total number. The selected number anchor points for each training set is 20 distributed in different areas of the ROI for both a short-distance AR session (10m distance) and a long-distance one (300m distance). In case of original SLAM, the input frame is not masked. The proposed pose estimation refinement achieves a better score than vSLAM on large-scale scenes and the relative distance values remain stable for a 50 sec period of tracking for both AR cases. The overall approximation accuracy needs yet to be improved.

Table 2. Reprojection RMSE error as a ratio for various anchor point selection sets

Case	Point Sets	Proposed method (%)	Original SLAM (%)
Short distance AR	1	0.66	0.91
	2	0.75	0.93
Long distance AR	3	0.74	0.52
	4	0.71	0.67

Benchmarking on detection and segmentation precision of MASK R-CNN was performed under different scale, rotation, distance and lighting conditions. A coloured bounding box and its confidence value (probability in percentage) are drawn around the ROI. In Figure 6, characteristic true positives are depicted with white while false positives (FP) and false negative (FN) results appear with red. FP concerns a misplaced detection and FN an undetected bounding box. It is indicated that misclassification occurs when the ROI covers a large area. This is due to the differences in rock formation of some instances (e.g., where the area is large and planar), accuracy of MASK R-CNN shape extraction as well as orientation in the camera frames. This is evident even in examples where ROIs are successfully predicted, however, the outputs tend to be misaligned towards the edges of the rock. So, the capturing of more representative samples, and especially through camera angles that showcase large ROIs, is necessary to further improve the performance. There are also, cases of false positive identifications of ROIs in vertical surfaces. An additional step of filtering these masks based on their orientation can quickly and effectively disregard such instances. However, the proposed method manages to detect and localize with high confidence even at 500 meters distance.

The influence of the ROI's physical distance on the positional error and the number of feature points detected was tested against a total number of 9 AR scenarios. It is assumed that features are correctly detected, already identified and their actual positions are computed independently. The camera looks at the ROI at a fixed vertical tilt angle $\theta^{\circ} = 35^{\circ}$ and the angles around the x and z-axis are constant at 0°. In every sequence that the distance of the camera to the pattern is changed, the center of the ROI is always projected in the center of camera frame. Table 3 shows the relationship between the distance from the camera and resulting pixel-size diameter of the detected ROI in the image frame. It also, presents the number of keypoints extracted by the detection algorithm of SLAM in the last frame of each sequence, as it is suggested by Mozos et al. (2007). This way prevents outliers or points that are lost during tracking, to be selected.

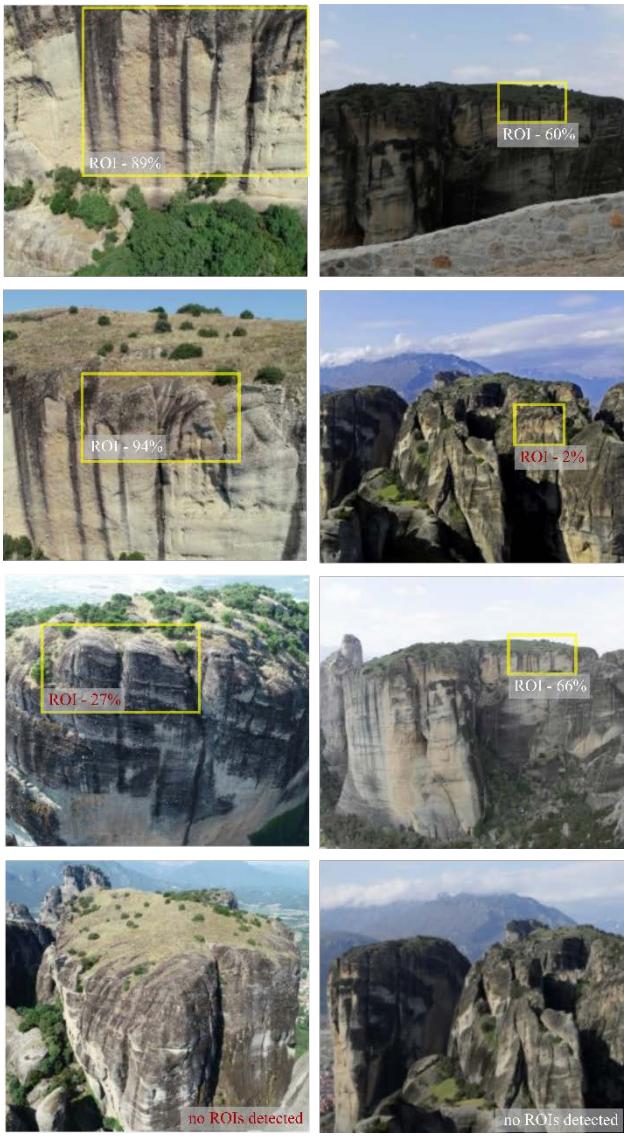

Figure 6. Output of the MASK-R CNN detector under different conditions of scale, rotation, lighting and occlusion: Bounding boxes out of the mask shapes with confidence value. Wrong predictions are highlighted with red.

Table 3. The number of feature points detected in the last frame of each sequence for different camera distances. The focal length is 5.58 mm for a resolution of 1080 x 2400 pixels.

Distance from camera to the ROI (meters)	Diameter (pixels)	Feature points (proposed method)	Feature points (original SLAM)
10	2143	747	2841
50	1758	650	1936
100	1181	513	1009
150	1026	477	520
200	778	462	499
250	542	365	176
300	209	387	93
400	136	288	74
500	98	141	89

Traditional SLAM gets more accurate when the size of the ROI on the image becomes bigger. Moreover, the positional error of Table 2 increases more than linearly with the distance, as it is expected. The proposed method outperforms the conventional SLAM only after 200 meters distance and yields approximately the same number of features in the tests of long-distance sequences. This observation can be explained by the construction of pixel-wise masks for non-relative objects in the image that keeps the detector running at the same but limited by the bounding box, region. Slower execution times, the threshold used for the validation of pose as well as the small dimensions of the projected ROI are the main factors resulting into the low number of features detected during short-distance AR.

The AR activity is being recorded five times with Android Profiler and its trace system for benchmarking validation. A seamless AR experience requires a stable frame rate of 30-40 frames per second (FPS), which equals to a 44.45 msec duration of capturing, processing, rendering and visualizing. Thus, a fixed framerate of 40 FPS is set. If rendering time exceeds this limit, frame drop occurs. Table 4 reports frame counts and latency at three critical moments: on pose estimation, on 3D model loading and on final display of 3D model, while Figure 7 demonstrates its CPU usage during the camera thread activity. The prototype succeeds in delivering a good frame duration distribution and maintaining the GPU at a consistent performance. A limitation of our method is indicated during short distance overlays due to the time spend during MASK-R CNN execution. However, the improvement in speed under far-field AR, in which the 3D model is superimposed 500 meters away from the camera device, is noticeable. It decreases tracking time up 42% and 3D models' drawing time to 32%. To investigate the degraded frame rate and latency of original vSLAM version, the CPU usage during a 15 sec long-distance AR session was captured (Figure 7).

Table 4. Metrics on UI thread, namely the main thread, at pose estimation, loading and drawn time.

Case	Process	Proposed method		Original SLAM	
		FPS	Delay	FPS	Delay
Short distance AR	Pose estimation	37	0.3 sec	40	0.1 sec
	Loading	35	3.8 sec	36	2.2 sec
	Display	33		36	
Long distance AR	Pose estimation	29	2.1 sec	23	3.6 sec
	Loading	27	4.5 sec	20	7.9 sec
	Display	26		20	

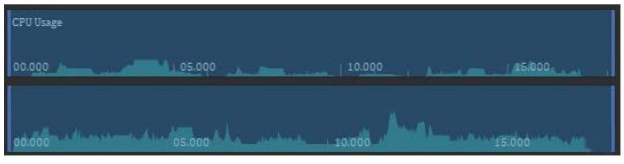

Figure 7. CPU usage comparison of our method (top) and original vSLAM (bottom), during a 15 sec AR session.

As it can be observed, the synergy of pose estimation refinement and mask overlay layer is computational effective with up to 25% decrease in CPU usage. The segmentation phase minimizes outliers and noise during SLAM's feature detection and yields to a noticeable increase in performance as the CPU computational overhead is minimized by 18%. The strategy of a fixed 40 FPS framerate and the exploitation of GPU acceleration during real-time processing compensate for accuracy and rendering speed.

6. CONCLUSIONS AND FUTURE WORK

The objects in the virtual and real scene must be precisely aligned with respect to each other in order to achieve the perception of coherent coexistence. Thus, pose estimation stage determines the quality, performance and immersion of the whole AR experience. DL and specifically, CNN models can be used as a means of optimizing this task. The proposed method enhances real-time large-scale outdoors tracking with MASK-R CNN instance segmentation and vSLAM of ARCore SDK. The Mask-R CNN model is successfully trained to distinguish between the rock cliff and its top surface while it excludes vegetation, build-up area and sky. During the AR session, it recognizes and localizes the ROI in the camera preview. The phase of segmentation creates a mask over other irrelevant regions and outputs the ROI ground-truth bounding boxes. Their calculated 2D-3D matches and the application of mask texturing over the adjacent regions in live camera frame, improve vSLAM's tracking accuracy and speed for far-field overlays. The prototype validates the applicability of the developed method in outdoors AR scenes of complex physical and geological characteristics. Currently, the AR tracking is not yet well-optimized for short-distance scenarios in which ARCore's Anchor functionality and SLAM consistently outperforms. Further work includes the adjustment of Mask-R CNN parameters and training settings to generalize to unseen complex natural scenes, optimization of spatial tracking accuracy as well as occlusion handling leveraging SLAM's scene mapping and depth perception. The impact of additional parameters like lighting conditions, camera distortion, occlusion and noise is necessary to be analysed. Finally, a similar evaluation needs to be undertaken for the IMU tracking errors accumulated over time in relevance to translation, rotation and distance from the camera.

ACKNOWLEDGEMENTS

This research has been co-financed by the European Union and Greek national funds through the Operational Program Competiveness, Entrepreneurship and Innovation, under the call RESEARCH–CREATE–INNOVATE (project code: T1EΔK-02859).

REFERENCES

Alzubaidi, L., Zhang, J., Humaidi, A.J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M.A., Al-Amidie, M., Farhan, L., 2021. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *J Big Data* 8, 53.

ARCore SDK Google Inc., 2018. ARCore Software Development Kit Software, Version 1.29. https://developers.google.com/ar (30 December 2021).

ARKit SDK Apple Inc., 2017. ARKit Software Development Kit Software, Version 5. https://developer.apple.com/augmented-reality (30 December 2021).

Baruah, A., Dev, A., Das, J., Misra, S.K., 2021. Android-Based Assistance System for Visually Impaired Person Using Deep Learning and Augmented Reality. *Chakraborty, M., Jha, R.Kr., Balas, V.E., Sur, S.N., Kandar, D. (Eds.), Trends in Wireless*

Communication and Information Security, Lecture Notes in Electrical Engineering. Springer Singapore, 175–186.

Bakalos, N., Soile, S., Ioannidis, C., 2020. Semantic classification of monuments' decoration materials using convolutional neural networks: a case study for meteora byzantine churches. *13th ACM International Conference on PErvasive Technologies Related to Assistive* Environments, 1-6.

Bassyouni, Z., Elhajj, I.H., 2021. Augmented Reality Meets Artificial Intelligence in Robotics: A Systematic Review. *Front. Robot. AI* 8, 724798.

Durrant-Whyte, H., Bailey, T., 2006. Simultaneous localization and mapping: part I. *IEEE Robot. Automat. Mag*. 13, 99–110.

EasyAR Sense, VisionStar Information Technology (Shanghai) Co, 2015. EasyAR Sense Software, Version 4.3. https://www.easyar.com/ (30 December 2021).
Freeman, J., 2020. Content enhancement with augmented reality and machine learning. *JSHESS* 70, 143.

Girshick, R., 2015. Fast R-CNN.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation.

He, K., Gkioxari, G., Dollar, P., Girshick, R., 2017. Mask R-CNN, *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, 2980–2988.

Jinyu, L., Bangbang, Y., Danpeng, C., Nan, W., Guofeng, Z., Hujun, B., 2019. Survey and evaluation of monocular visual-inertial SLAM algorithms for augmented reality. *Virtual Reality & Intelligent Hardware*. 1, 386–410.

Keras Keras Development Team, 2015. Keras Software, Version 2.4. Chollet, F. & others. https://github.com/fchollet/keras/ (30 December 2021).

Le, H., Nguyen, M., Yan, W.Q., Nguyen, H., 2021. Augmented Reality and Machine Learning Incorporation Using YOLOv3 and ARKit. *Applied Sciences* 11, 6006.

Lee, Suwoong, Lee, Seungjae, Lee, K., Ko, J.G., 2019. DeepMobileAR: A Mobile Augmented Reality Application with integrated Visual SLAM and Object Detection, *SA '19: SIGGRAPH Asia 2019*, ACM, Brisbane QLD Australia, 1–2.

Liu, L., Li, H., Gruteser, M., 2019. Edge Assisted Real-time Object Detection for Mobile Augmented Reality. *MobiCom '19: The 25th Annual International Conference on Mobile Computing and Networking*, ACM, Los Cabos Mexico, pp. 1–16.

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

MLKit Google Inc., 2021. Machine Learning Kit Software, Version 16.2.1. https://developers.google.com/ml-kit (30 December 2021).
Mousavian, A., Anguelov, D., Flynn, J., Kosecka, J., 2017. 3D Bounding Box Estimation Using Deep Learning and Geometry. arXiv:1612.00496 [cs].
Mozos, Ó.M., Gil, A., Ballesta, M., Reinoso, O., 2007. Interest Point Detectors for Visual SLAM, in: Borrajo, D., Castillo, L., Corchado, J.M. (Eds.), Current Topics in Artificial Intelligence, Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin, Heidelberg, 170–179.

MXNet Apache Team, 2017. Apache MXNet Software, Version 1.9. Apache Software Foundation. https://mxnet.apache.org/versions/1.9.0 (30 December 2021).

Nister, D., Naroditsky, O., Bergen, J., 2004. Visual odometry. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004.* CVPR 2004., IEEE, Washington, DC, USA, 652–659.

Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc.; 8024–8035.

Pauly, O., Diotte, B., Fallavollita, P., Weidert, S., Euler, E., Navab, N., 2015. Machine learning-based augmented reality for improved surgical scene understanding. *Computerized Medical Imaging and Graphics* 41, 55–60.

Perdpunya, T., Nuchitprasitchai, S., Boonrawd, P., 2021. Augmented Reality with Mask R-CNN (ARR-CNN) inspection for Intelligent Manufacturing, *The 12th International Conference on Advances in Information Technology*, ACM, Bangkok Thailand, 1–7.

Rao, J., Qiao, Y., Ren, F., Wang, J., Du, Q., 2017. A Mobile Outdoor Augmented Reality Method Combining Deep Learning Object Detection and Spatial Relationships for Geovisualization. *Sensors*, 17, 1951.

Sahu, C.K., Young, C., Rai, R., 2021. Artificial intelligence (AI) in augmented reality (AR)-assisted manufacturing applications: a review. *International Journal of Production Research 59*, 4903–4959.

Su, Y., Rambach, J., Pagani, A., Stricker, D., 2021. SynPo-Net—Accurate and Fast CNN-Based 6DoF Object Pose Estimation Using Synthetic Training. *Sensors* 21, 300.

TensorFlow Lite Google Inc., 2015. TensorFlow Lite Software, Version 2.7. https://www.tensorflow.org/lite (30 December 2021).

Vuforia SDK PTC Inc., 2015. Vuforia Augmented Reality Software Development Kit Software, Version 10.3. https://developer.vuforia.com/ (30 December 2021).

Wang, J., Wang, Q., Saeed, U., 2018. A visual-GPS fusion based outdoor augmented reality method. *VRCAI '18: International Conference on Virtual Reality Continuum and its Applications in Industry*, ACM, Tokyo Japan, 1–4.

WikiTube Qualcomm Inc., 2008. Wikitube Software, Version 8.2. https://www.wikitude.com/ (30 December 2021).

Zhang, J., Cosma, G., Watkins, J., 2021. Image Enhanced Mask R-CNN: A Deep Learning Pipeline with New Evaluation Measures for Wind Turbine Blade Defect Detection and Classification. *J. Imaging* 7, 46.