Supplementary Material: A 2D-3D Object Detection System for Updating Building Information Models with Mobile Robots

Max Ferguson
Stanford University
maxferg@stanford.edu

Kincho Law
Stanford University
law@stanford.edu

Abstract

The supplementary material contains: (1) a video that shows detailed and comprehensive results, as well as a PDF file which includes (2) more experimental results and examples, (3) additional implementation details and a description of the 3D convolutional neural network (4), additional information about the worksite object dataset that was collected and annotated for this project, and (5) a discussion of potential applications that extend beyond the evident utility of the system.

1. Additional Results

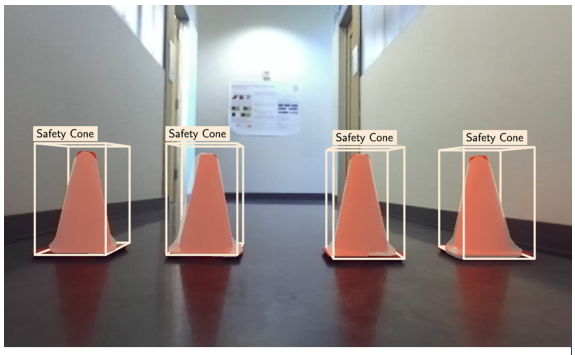
In this section, we present additional results that demonstrate how our object detection and tracking system is used to spawn objects in an existing geometric model of the building environment.

1.1. Comparing Mask R-CNN and Object R-CNN

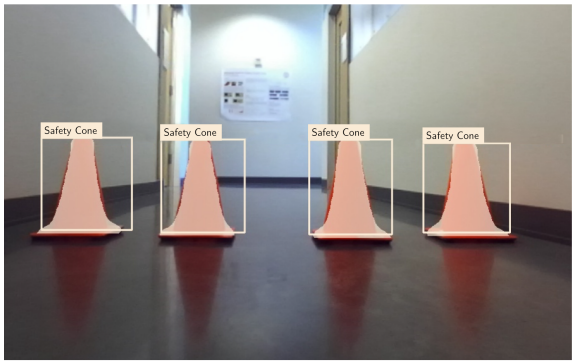
We consider Object R-CNN as an extension of the 2D instance segmentation framework, Mask R-CNN. Figure S1 compares the output of Mask R-CNN and Object R-CNN on an image from our worksite dataset. The object class and segmentation predictions are very similar, as both frameworks use the same architecture for class prediction and segmentation. However, unlike Mask R-CNN, Object R-CNN also generates 3D bounding boxes. In frame-rate critical applications, Object R-CNN could use the Faster R-CNN architecture in place of Mask R-CNN. However, when testing Object R-CNN with Faster R-CNN we observed a minor reduction ($\sim$1 % mAP) in 2D bounding box prediction accuracy.

1.2. Filtering False-positive Detections

The proposed multi-object tracking system provides an effective method of filtering false-positive detections, as discussed in the paper. To demonstrate the benefits of the object tracking system, the proposed system was tested with the tracking system enabled and disabled. When the tracking system was disabled, objects were added to the building information model immediately after being detected by Object R-CNN. Figure S2 compares the output of the system in these two tests. As shown in Figure S2, noise in the object detection and mobile robot localization process causes objects to be spawned incorrectly in the geometric model. By tracking objects across multiple frames, the negative impact

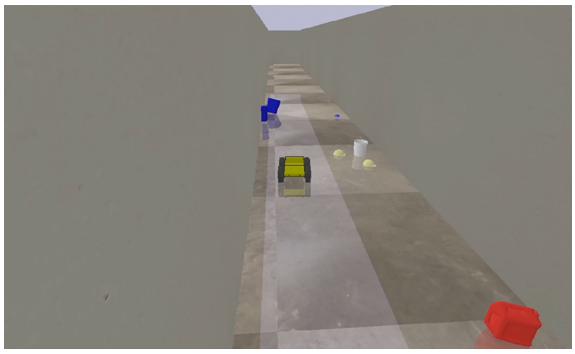


Object R-CNN (Ours)

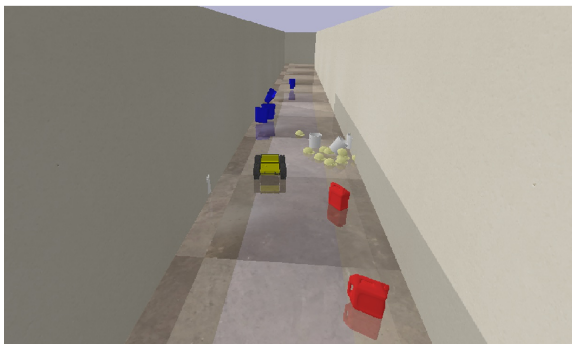


Mask R-CNN

Figure S1. Comparison of output from the 2D instance segmentation network Mask R-CNN and our 2D-3D object detection network, Object R-CNN.



Proposed System with Tracking / Filtering



Without Tracking / Filtering

Figure S2. Comparison of the updated building information model with the tracking system enabled and disabled.

of this noise is significantly reduced.

1.3. Object Tracking Hyperparameters

Objects are only added to the building information model after being tracked successfully for a certain number of frames. In this section, we investigate how this number, k, effects the precision and recall of the system. A series of experiments are conducted using different values of k. The proposed system is used to dynamically update the building information model, as described in the paper. The results from these experiments are shown in Figure S3. The maximum F1 score is obtained with $k = 4$, and hence this value of k is used in all other experiments, including those described in the paper.

2. Additional Model Details

In this section, we provide additional details about the Object R-CNN architecture and training process.

2.1. Mask R-CNN Backbone

When selecting the Mask R-CNN backbone we choose to prioritize accuracy over training and evaluation speed.

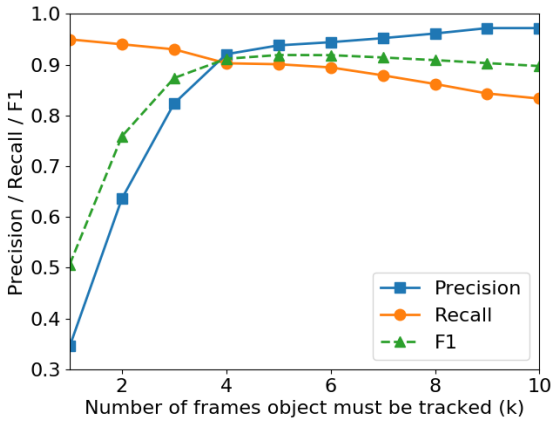


Figure S3. Scene reconstruction accuracy when varying the number of frames (k) that a worksite object must be tracked for before it is added to the building information model.

We use the Inception-ResNet-V2 architecture with Atrous convolutions, as the Mask R-CNN backbone [3]. Furthermore, we use a feature pyramid network (FPN) to extract features at different scales from the backbone, as described in [1].

2.2. Neural Network Architecture

In Object R-CNN, a 3D CNN is used to extract features from a 3D voxel representation of each detected worksite object. For the sake of simplicity, we use a CNN architecture similar to VoxNet [2]. This architecture has alternating convolution and pooling layers to gradually build up a rich feature representation of the object. The input size is chosen to be $64 \times 64 \times 64$. That is, we convert the points inside each 3D anchor to a $64 \times 64 \times 64$ voxel representation, and pass that representation to the 3D CNN. The architecture of the 3D CNN is described in Table S1.

2.3. Training

In this section, we provide some additional information about the Object R-CNN training process. All RGB-D images are resized such that their scale (shorter edge) is 800 pixels. To improve training speed, the 3D CNN is only trained on the top 100 region proposals proposals from each image, ordered by objectness score. When training, each 2D region proposal from the region proposal network (RPN) is projected into the point cloud to obtain the voxel inputs for training. When evaluating the network, a slightly different procedure is used. The trained Mask R-CNN network is used to generate 2D bounding boxes and segmentation masks. Each 2D bounding box (detection) is then projected into the point cloud and used to obtain 3D features for size and position predictions.

Block	Layer Type	Filter Size $(w \times h \times d, n)$	Output Size $(w \times h \times d \times channels)$
Input	Cropped Voxel Input	-	64×64×64×4
Conv Block 1	Convolution	3×3×3, 8	64×64×64×8
	ReLU	-	64×64×64×8
	Max Pooling	-	32×32×32×8
Conv Block 2	Convolution	3×3×3, 16	32×32×32×8
	ReLU	-	32×32×32×16
	Max Pooling	-	16×16×16×16
Conv Block 3	Convolution	3×3×3, 16	16×16×16×16
	ReLU	-	16×16×16×16
	Max Pooling	-	8×8×8×16
Conv Block 4	Convolution	3×3×3, 32	8×8×8×32
	ReLU	-	8×8×8×32
	Max Pooling	-	4×4×4×32
Mask R-CNN Concat	Flatten	-	2048
	Concatenation	-	3072
Output	Dense	-	1024
	ReLU (size), tanh (pos)	-	6

Table S1. Architecture of the convolutional neural network used for size and position prediction. Zero-padding is used in all convolution layers to ensure that the input dimensions are unchanged. Features from Mask R-CNN are introduced into the network in the "Concatenation" layer. The output of the neural network is a 6 element vector describing the position and size of the object in the Region of Interest (RoI).

3. Dataset

In this section, we provide additional details about the worksite object dataset. The dataset consists of multiple sequences of RGB-D images captured by a purpose-built mobile robot. The RGB-D images were captured using a Zed Stereo camera. Each image has resolution 1920×1080 pixels. The position and orientation of the mobile robot is provided for each frame. Visual odometry and Lidar were both used to infer the position of the robot in the building. Cubic interpolation was used to infer the position and orientation of the robot at the exact time each RGB-D frame was captured, as the localization algorithm only provided the location at discrete points in time.

Objects in the RGB-D dataset were annotated with 2D segmentation masks and 3D bounding boxes. 2D bounding boxes were automatically generated from the segmentation masks. Segmentation masks were labelled as polygon regions, and subsequently converted to bitmap masks. The segmentation masks do no extend beyond the bounds of the image. In the case where an object is occluded, the segmentation mask only includes the visible pixels. When annotating objects with 3D bounding boxes, we aimed to create a box that tightly enclosed the true object. For this reason, the 3D boxes may extend beyond the limits of the 2D image. In the case that the object is partially occluded, the 3D bounding box should still enclose the entire object.

4. Potential Applications

The primary focus of this work was produce a system that could automatically detect newly placed objects in a large indoor space. One potential application of this technology is automatically obtaining statistics about objects in a facility-scale space. This could be beneficial in the management of large-scale facilities or worksites.

A second application is detecting hazardous objects for the purpose of worksite safety. That is, the proposed system could be used to detect certain groups of objects which may be hazardous together, or objects which are hazardous in certain positions.

Finally, this work represents a step forward in semantic understanding of large-scale worksite facilities. Semantic understanding is a critical factor in the development of autonomous worksite robots. Hence, we expect that future robotic systems will collaboratively build dynamic and semantic-rich models of the environment using a technique similar to the one described in this work.

References

[1] T.-Y. Lin, P. Dollr, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[2] D. Maturana and S. Scherer. VoxNet: A 3D convolutional neural network for real-time object recognition. In *IEEE/RSJ*

International Conference on Intelligent Robots and Systems (IROS), pages 922–928, 2015.

[3] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, Inception-ResNet and the impact of residual connections on learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.