1 Introduction

The paper tests the feasibility of domain randomization in bridging the "reality gap". The paper focuses on image-based tasks and proposes to randomize the rendering of images in simulation. The paper tests its approach on two tasks, object-localization and using the pre-trained object-localization network as a feature extractor for an off-the-shelf motion planning package to perform object picking. The paper argues that testing on object-localization makes sense because it serves as stepping stone to more general robotic manipulation skill.

2 Legends

1. Text followed by (?) indicates that the paper does not explain clearly the point in the text.

3 Object Localization

3.1 Task Description

Given an object of interest s_i, our goal is to train an object detector $d(I_0)$ that maps a single monocular camera frame I_0 to the x-y coordinate of the object. The object s_i can be one of cone, cube, cylinder, hexagonal prism, pyramid, rectangular prism, tetrahedron, and triangular prism. A network is trained for each possible object. The object is placed on a table top. There are three possible scenarios:

- Only the object of interest is present.

- So-called distractor objects, also taken from the sets of available objects, are placed near the object of interest.

- Distractor object partially occludes the object of interest.

To evaluate the accuracy of a network in the real world, a test set is created that consists of images from the real world. Each possible object has 60 images, with 20 images in each possible scenarios. The distance between the object and the camera ranges from 70cm to 105cm. The camera position remains constant across all images. They do not control for lightning conditions or the rest of the scene around the table (all images contain part of the robot and tape and wires on the floor).

The ground truth position for each object in each image is measured by aligning the object on a grid on the tabletop (?).

3.2 Domain randomization technique

The following aspects are randomized for each sample used during training:

- Number and shape of distractor objects on the table: between 0 and 10 distractor objects are added to the table in each scene.

- Position and texture of all objects on the table: random textures are chosen among the following: (a) A random RGB value. (b) A gradient between two random RGB values. (?) (c) A checker pattern between two random RGB values. (?). The textures of all objects are chosen uniformly at random.

- Textures of the table, floor, skybox (?), and robot.

- Position, orientation and field of view of the camera. The camera in the simulated scene is manually placed to approximately matches the viewpoint and field of view of the real camera. Each training sample places the camera randomly within a $(10 \times 5 \times 10)$ cm box around this initial point. The viewing angle of the camera is calculated analytically to point at a fixed point on the table, and then offset by up to 0.1 radians in each direction.

The field of view is also scaled by up to 5% from the starting point.

- Number of lights in the scene (?).

- Position, orientation, and specular characteristics of the lights (?).

- Type and amount of random noise added to images (?).

The detector does not have access to the color of the object of interest at training time, only their size and shape (?). MuJoCo is used as the renderer. The height of the table is fixed during simulation. They construct mesh representations for each object to render in the simulator.

4 Architecture and training details

The VGG-16 architecture is used. They use the standard VGG convolutional layers, and use smaller fully connected layers of sizes 256 and 64 and do not use dropout.

The detector is trained with Adam to minimize the L_2 loss between the object positions estimated by the network and the true object position. They found that using a learning rate of around $1e-4$ rather than $1e-3$ improved convergence and helped avoid a common local optimal optimum, mapping all objects to the center of the table.

Each training sample consists of: (a) a render image of the object of interest and one or more distractors on a simulated tabletop (contradiction, previously mentioned 0 to 10 distractor objects are added) (b) a label corresponding to the Cartesian coordinates of the center of the mass of the object in the world frame.