Robot Learning from Human Demonstration in Virtual Reality

Extended Abstract

Francesca Stramandinoli[1], Kin Gwn Lore[1], Jeffrey R. Peters[1], Paul C. O'Neill[1], Binu M. Nair[2], Richa Varma[2], Julian C. Ryde[2], Jay T. Miller[1], Kishore K. Reddy[1]

[1]United Technologies Research Center, 411 Silver Lane, East Hartford, Connecticut, 06108, USA

[2]United Technologies Research Center, 2855 Telegraph Avenue, Berkeley, California, 94705, USA

{stramaf,lorek,petersjr,oneillpc,nairbm,varmar,rydejulc,millertjt,reddykk}@utrc.utc.com

ABSTRACT

To best leverage the adoption of robotics systems, their capability to learn and adapt to novel situations is essential. We propose an immersive Virtual Reality (VR) environment in which a simulated robot is trained through human demonstration; the VR environment is leveraged to communicate to the robot the intent of a task, so that the robot can then replicate it in a broader set of initial conditions. The skills acquired by the simulated robot are then transferred to the real robot in the physical world. The approach is intuitive, safe for the human demonstrator and reduces the downtime of the robot during the training process. We demonstrate this proposed framework for a pick and pass operation.

KEYWORDS

Human-Robot Interaction, Virtual Reality Intention Communication, Learning from Demonstration.

ACM Reference Format:
Francesca Stramandinoli[1], Kin Gwn Lore[1], Jeffrey R. Peters[1], Paul C. O'Neill[1], Binu M. Nair[2], Richa Varma[2], Julian C. Ryde[2], Jay T. Miller[1], Kishore K. Reddy[1]. 2018. Robot Learning from Human Demonstration in Virtual Reality: Extended Abstract. In *Proceedings of Workshop on Virtual, Augmented and Mixed Reality for Human-Robot Interaction (HRI'18).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 BACKGROUND

In the last decade, robotics systems have become progressively more present in different application domains ranging from inspection and maintenance [11], to surveillance and security [17], to manufacturing [10]. To increase the adoption of robotics systems, their capability to learn and adapt to novel situations is essential.

Robotics systems can learn new skills either autonomously or through human interaction. Robots that learn autonomously, typically through trial and error, try to discover the best way to perform a certain task (i.e. reinforcement learning [8]). However, this approach can be time consuming and requires a large number of roll-outs or trials. Training through human interaction can occur through different modalities (e.g. speech and/or demonstration via gestures). Interaction through speech offers a natural interface for exchanging information about a task within the involved partners. Unfortunately, many tasks can be too difficult to explain succinctly in words. Training a robot through human demonstration via gesture has several advantages: (i) it represents an intuitive interface for humans, introducing an abstraction layer from the robot programming language, (ii) it provides clues on the motor skills needed to perform a task (e.g. tracking the human motion while performing the task and imitating it), and (iii) it leads to a faster acquisition of skills, avoiding random exploration of all possible ways of performing a task via trial and error.

The training of a robot can be performed through demonstration on the real platform by an observed human demonstrator, an instrumented secondary robotic platform, or in a simulated environment (see [1] for an excellent review of learning from demonstration outside of simulation). Robotic training that is performed in a simulated environment is advantageous, since it is not dependent on the availability of a physical robotic platform, reducing the downtime needed for training and testing new skills on the real platform. Indeed, simulated environments can allow several instances of the training/testing to be run simultaneously, enabling fast comparisons of design choices, including parallelism, network architectures, and other training related parameters. Moreover, the human teacher doesn't need to share the workspace (e.g. hostile environment) with the real robot, reducing the exposure of human personnel to safety issues. This paradigm allows for the human and robot to both perform the task in their own embodiment (each with their own joint configuration).

We propose a Virtual Reality (VR) environment as a means to intuitively and safely communicate to a virtual robot the intent of a task, so that the robot can then replicate it in a broader set of initial conditions. A similar approach for communicating the intent of a task to a robot has been adopted in [6] where the authors proposed a framework for one-shot imitation learning. They considered a block stacking task with a varying number of blocks and stacking arrangements, with the number of blocks allowed to vary between 2 and 10. They collected 1,000 trajectories for 140 training tasks through a hard-coded policy and 43 test tasks, testing their one-shot imitation learning on both training and test tasks with good results. Through the one-shot imitation learning setting they proposed, a robot could learn from a single demonstration of a possible instantiation of the block stacking task and instantly replicate the new task with a different set of initial block configurations.

Here we focus on a VR environment to demonstrate grasping of common objects with a robotic arm. A perception pipeline was built to recognize these objects and calculate their orientation on a table. The demonstration learnings were then used to successfully pick and pass real objects selected by an operator using a voice interface to communicate with the physical robot.

2 VR ENVIRONMENT FOR ROBOT TRAINING

The proposed VR environment consists of the following main components:

- A physics engine that can perform forward dynamics simulation, inverse dynamics computation, forward and inverse kinematics and collision detection for articulated bodies loaded in a simulated environment.
- 3D models of the adapted robotic platform and of the objects that the robot needs to interact with. These models are loaded and simulated in the physics engine.
- A VR interface that enables the interaction of a human with the robot and objects simulated in the physics engine. The VR interface includes a commercial off the shelf VR headset for receiving visual feedback and two hand controllers for interacting in the VR environment and guide the movement of the virtual robotic platform. Through the VR interface, the human can observe the environment and train the virtual robot by providing demonstrations of the task in different environment conditions.

In our setup, we have adopted Bullet Physics as the physics engine [15], and leveraged PyBullet (version 2.86) to interface a HTC Vive controller [5] with the simulated environment. This setup is fully documented and supported by the tool, and requires a limited effort to be implemented on a workstation.

The adaptation of this environment for a specific application requires modelling the training scenario. Bullet physics supports the Unified Robot Description Format (URDF), a standard XML format for describing the model of a robot [7, 12]. URDFs, typically made available by robot vendors, can include the physical geometry, the kinematic and dynamic properties of joints and links, and even sensor locations and properties on a robot. Moreover, it is possible to find URDF model libraries of tools and objects that might be required or involved in the robot operation, and are hence needed for the training. In our setup we leveraged the Yale-CMU-Berkeley (YCB) object and model set [4] as well as models from the robot arm manufacturer, Kinova [13]. Therefore, the modeling effort is limited to the description of the overall simulation scene, positioning the robot, designing the working area and placing tools and objects according to the training scenario to explore.

3 APPLICATION AND EVALUATION

The VR environment has been applied to train a robotic arm for a pick and pass application (Fig. 1). In this scenario, the robot has the role of an assistant to a human operator involved in a maintenance task of mechanical equipment (Fig. 2). Upon a vocal request by the human, the robot identifies the requested tool (out of the set of phillips screwdriver, flathead screwdriver, clamp, wrench, scissors, marker, hammer or drill), picks it from the workspace and hands it to the human.

3.1 Setup

Figure 3 shows the architecture of the robotic system. The robotic platform adopted for this application is the Kinova Mico2 arm

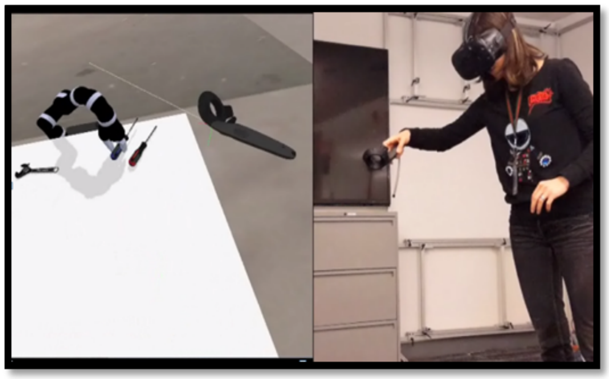


Figure 1: VR environment leveraged by the human operator to demonstrate to the virtual robot different grasp configuration for interacting with tools placed on the table. Video of a demonstration of the VR environment for robot training available upon request to authors.

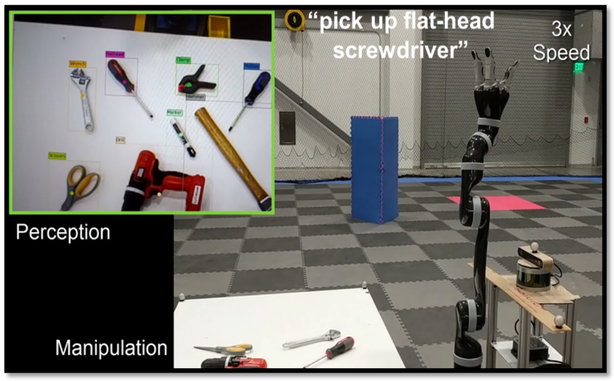


Figure 2: The Mico2 robotic arm has the role of an assistant of a human operator involved in a maintenance task of mechanical equipment. Video of a demonstration of the robot picking and passing tools to a human operator available upon request to authors.

with 6 degrees of freedom [13] and a three finger under-actuated manipulator. The robot arm is oriented vertically, mounted to a base. An Android application running on a tablet provides a voice-based interface to receive commands from the human operator. A manipulation planner controls the sequencing of the pick and pass operation. The system uses an ASUS Xtion RGBD camera [2] to provide vision of the working environment. A perception pipeline uses the images captured by the camera to identify and localize (i.e. determine position and orientation) of a required tool in the workspace while the grasp configuration module determines the best grasp configuration for a specific tool, according to its shape and orientation on workspace.

All the modules are integrated using the Robot Operating System (ROS) middleware [12]. During operation, the user command is parsed by the Android application. This passes the received information to the manipulation planner that interrogates the perception

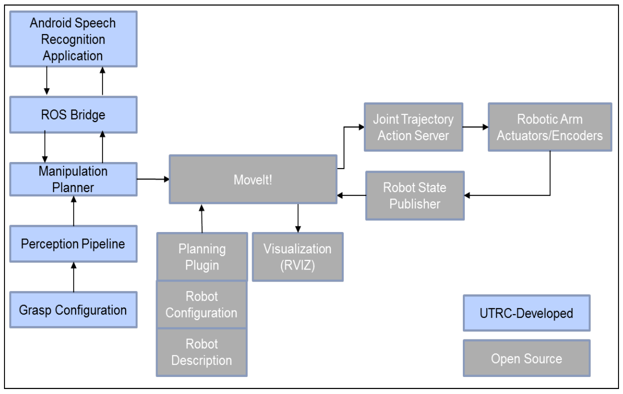


Figure 3: Illustration of the architecture of the robotic system.

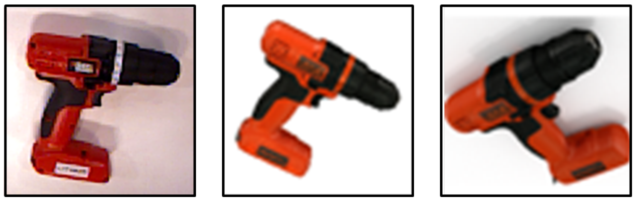
pipeline to identify and localize the requested tool, and then references the grasp configuration module to determine the best grasp configuration. The manipulation planner then controls the robot in the execution of the pick and pass operation, with operations outside of the grasp orientations planned and controlled using MoveIt! [18].

3.2 Robot training

The training of the robotic system involves the perception pipeline and the grasp configuration. The perception pipeline is trained to classify a set of tools, and to determine their position and orientation in the workspace. For the classification of tools we leveraged on a Convolutional Neural Network (CNN) [9] that receives a set of images of the tools to be manipulated by the robot as an input. Typically, supervised machine learning methods need large quantities of labeled training data to achieve high accuracy. Options to generate this data vary in fidelity, level of effort for labeling, and processing time, as summarized here:

- Real data with hand labeling, 0.1Hz
- Photo-realistic rendered data with CPU or accelerated GPU techniques, 1Hz
- Rendered data with GPU, 100Hz

Conventional real-time rendering with the GPU is designed for human interaction and therefore needs to be at least 24Hz and so makes approximations to accommodate high frame rates. Shortcuts include discarding geometry outside the field of view, ignoring indirect illumination effects, and often simplifying or discounting ambient occlusion. However, these effects are important for understanding scenes in the context of extracting 3D shape (e.g. shape from shading approaches [21]). Ambient occlusion is particularly important for inferring scene and object structure in the absence of strong texture variation, such as in the cases of smooth-walled featureless interiors and snow covered environments. Three dimensional shape, rather than image appearance alone, is vital for recognizing objects for grasping and manipulation affordance and for establishing the object's position and orientation. The aforementioned approximations for real-time rendering with the GPU result in images that cannot be mistaken for real images, despite allowing



(a) Real Image (b) Low Fidelity Image (c) High Fidelity Image

Figure 4: Dataset samples collected from different sources used to train the classifier.

Table 1: Evaluation of the classification accuracy after training the CNN with a specific training set (i.e. images taken from real camera, low fidelity synthetic images and high fidelity synthetic images) against the same test set collected from the real camera.

	Test Accuracy
Real	98.80%
Low Fidelity	83.13%
High Fidelity	93.96%

real-time operation and generating images that enable the human users to visualize the 3D world to some degree. Photo-realistic rendering generates images that are virtually indistinguishable from real scenes, particularly for movie special effects, art, architectural and other visualizations. They make far fewer approximations and render a scene by essentially sampling photon propagation paths in as unbiased a manner as possible. As such, it seems likely that if they are close to indistinguishable for humans, using these images to training deep neural networks should result in networks that rival human performance on image processing. Finally, we anticipate photo-realistic rendered training datasets will enable enhanced network generalizability and reduce the risk of over fitting. For a concrete comparison of CNNs trained with these different sources, we trained networks with images from the camera system used in the experimental setup (Fig. 4(a)), OpenGL [20] scenes rendered with TinyRenderer [16] (Fig. 4(b)), and scenes rendered with photo-realistic renderer Cycles in Blender [14] (Fig. 4(c)). Due to time limitations, none of simulated training sets included texture randomization, which has also been shown to improve performance of synthetic training data [19]. For each tool we collected 200 RGBD images in randomized positions, orientations, and lighting conditions. The CNN was trained for each dataset independently. In Table 1 we provide the evaluation of the classification accuracy after the training with a specific training set against the same test set collected from the real camera. Note that the photo-realistic images provided much higher success rates than the low-fidelity synthetic images, but were not quite as successful as a network trained on real images.

For localization, the perception pipeline leverages depth and RGB images and built-in OpenCV functions to perform segmentation and pose estimation of the tools [3].

The grasp configuration module is trained to identify the best grasp configuration for a tool according to its orientation. The best grasp configuration is the one ensuring the highest probability of success in the pick and pass operation. This training is performed in the VR environment by a human operator guiding the robot in the exploration of different grasp configurations, i.e. different positions and orientations of the end-effector of the robot with respect to the tool to grasp. Using the hand controller, the human trainer can guide a virtual robotic arm in a desired grasp configuration, and control the opening and closing operation of the end-effector. For each trial, the success or failure of the grasp attempt is recorded. The collected data are passed to a machine learning algorithm to extract a relation between the tool to grasp, the grasp configuration and the outcome of the grasp. The machine learning algorithm is therefore able to determine the best grasp configuration for a specific tool.

4 CONCLUSIONS

In this work, we presented an immersive Virtual Reality (VR) environment in which a simulated robot is trained through human demonstration. The operator can leverage the VR environment to communicate to the robot the intent of a task, so that the robot can then replicate it in a broader set of initial conditions. The skills acquired by the simulated robot are then transferred to the real robot in the physical world. The approach is intuitive, safe for the human demonstrator and reduces the downtime of the robot during the training process.

We plan to extend the current framework to enable grasp configurations with visual servoing (continuous visual feedback for pick up), which will allow more end-to-end learning without expecting a separate perception module. Moreover, in the current setup we constrain the gripper to a perpendicular orientation with respect to a flat table surface upon which the tools are laying. Grasps are then executed at a constant height such that the gripper is at a certain distance above the surface. The remaining parameters to be learned which define the grasp pose, are therefore the translational position on the surface, and the rotation of the gripper with respect to the surface's normal. We plan to remove this constraint to take full advantage of the degrees of freedom of the robotic arm and achieve a greater range of grasps to be executed.

ACKNOWLEDGMENTS

The authors would like to thank Soumalya Sarkar and Jerry Ding for their valuable comments and helpful suggestions.

REFERENCES

[1] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. 2009. A survey of robot learning from demonstration. *Robotics and autonomous systems* 57, 5 (2009), 469–483.

[2] ASUS. 2017. Technical specifications of ASUS Xtion RGBD camera. https://www.asus.com/us/3D-Sensor/Xtion_PRO_LIVE/specifications/. (2017).

[3] G. Bradski. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000).

[4] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. 2015. Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set. *IEEE Robotics & Automation Magazine* 22, 3 (2015), 36–52.

[5] HTC Corporation. 2016. The HTC Vive virtual reality headset. https://www.vive.com/us/product/vive-virtual-reality-system/. (2016).

[6] Yan Duan, Marcin Andrychowicz, Bradly Stadie, Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. 2017. One-shot imitation learning. In *Advances in neural information processing systems*. 1087–1098.

[7] Willow Garage. 2012. Xml robot description format (urdf). http://www.ros.org/wiki/urdf/XML. (2012).

[8] Jens Kober, J Andrew Bagnell, and Jan Peters. 2013. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* 32, 11 (2013), 1238–1274.

[9] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[10] S Mitsi, K-D Bouzakis, G Mansour, D Sagris, and G Maliaris. 2005. Off-line programming of an industrial robot for manufacturing. *The International Journal of Advanced Manufacturing Technology* 26, 3 (2005), 262–267.

[11] Nicolas Pouliot and Serge Montambault. 2008. Geometric design of the LineScout, a teleoperated robot for power line inspection and maintenance. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 3970–3977.

[12] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. 2009. ROS: an open-source Robot Operating System. In *ICRA workshop on open source software*, Vol. 3. Kobe, Japan, 5.

[13] Kinova Robotics. 2017. Technical specifications of Mico2 6 DOF robotic arm. http://www.kinovarobotics.com/wp-content/uploads/2015/02/Kinova-Specs-MICO2-6DOF-Web-170512-1.pdf. (2017).

[14] Ton Roosendaal and Stefano Selleri. 2004. *The Official Blender 2.3 guide: free 3D creation suite for modeling, animation, and rendering*. Vol. 3. No Starch Press San Francisco.

[15] Real-Time Physics Simulation. 2017. Documentation of Bullet Physics Library. http://bulletphysics.org/wordpress/?page_id=9. (2017).

[16] Dmitry V. Sokolov. 2016. How OpenGL works: software rendering in 500 lines of code. https://github.com/ssloy/tinyrenderer/wiki. (2016).

[17] Guangming Song, Kaijian Yin, Yaoxin Zhou, and Xiuzhen Cheng. 2009. A surveillance robot with hopping capabilities for home security. *IEEE Transactions on Consumer Electronics* 55, 4 (2009).

[18] Ioan A. Sucan and Sachin Chitta. 2015. MoveIt! [Online] http://moveit.ros.org. (2015).

[19] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. 2017. Domain randomization for transferring deep neural networks from simulation to the real world. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 23–30.

[20] Mason Woo, Jackie Neider, Tom Davis, and Dave Shreiner. 1999. *OpenGL programming guide: the official guide to learning OpenGL, version 1.2*. Addison-Wesley Longman Publishing Co., Inc.

[21] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah. 1999. Shape-from-shading: a survey. *IEEE transactions on pattern analysis and machine intelligence* 21, 8 (1999), 690–706.