Deep Learning of Augmented Reality based Human Interactions for Automating a Robot Team

Adhitha Dias
Department of Electronic and Telecommunication Engineering
University of Moratuwa
Colombo, Sri Lanka
adhithadias27@gmail.com

Hasitha Wellaboda
Department of Electronic and Telecommunication Engineering
University of Moratuwa
Colombo, Sri Lanka
hasithaprashan@gmail.com

Yasod Rasanka
Department of Electronic and Telecommunication Engineering
University of Moratuwa
Colombo, Sri Lanka
yasodrasanka@gmail.com

Menusha Munasinghe
Department of Electronic and Telecommunication Engineering
University of Moratuwa
Colombo, Sri Lanka
menushamilaj@gmail.com

Ranga Rodrigo
Department of Electronic and Telecommunication Engineering
University of Moratuwa
Colombo, Sri Lanka
ranga@uom.lk

Peshala Jayasekara
Department of Electronic and Telecommunication Engineering
University of Moratuwa
Colombo, Sri Lanka
peshala@uom.lk

***Abstract*— Getting a team of robots to achieve a relatively complex task using manual manipulation through augmented reality (AR) is interesting. However, the true potential of such an approach manifests when the system can learn from humans. We propose a system comprising a team of robots that performs a previously unseen task—a variant, to be specific—by learning from the sequences of actions taken by multiple human beings doing this task in various ways using deep learning (DL). The training inputs can be through actual manipulation of the team of robots using an augmented-reality tablet or through a simulator. Results indicate that the system is able to fulfill the specified variant of the task more than 80% of the time, inaccuracies mainly owing to unrealistic specifications of tasks. This opens up an avenue of training a team of robots, instead of crafting a rule base.**

***Keywords*— deep learning, robot collaboration, augmented reality, automate**

I. Introduction

It is intriguing to see the ease with which humans do relatively complex tasks comprising a sequence of steps. The sequences of steps themselves are not identical if several humans attempt the same complex task. Moreover, several variants of sequences can lead to successful completion of the task—a variant. E.g., in a building construction task, for a particular building design, there can be many sequences of steps of successfully building, while there are different building designs. In other words, each task can have many variants and each variant emanates via a number of sequences of steps. Getting a set of robots in a rule-based algorithmic system that seems to achieve the same level of competence is difficult to craft due to the variations [1]. In contrast, learning to do a relatively complex task by learning from multiple humans completing several variants of the task many a times, is an intriguing solution.

Calinon *et al*. [2] discusses about robot programming by demonstration for learning and reproduction of gestures using Hidden Markov Models and Gaussian Mixture Regression. Laskey *et al*. [3] compares human-centric and robot-centric sampling for robot deep learning from demonstrations. Michael *et al*. [4] performs multi-robot task assignment without relying on a priori assignment information.

However, none of these systems highlights the significance of using the output from DL to control task of a multi-robot system and none of these works have used AR for user interactions and demonstration purpose of the task.

In this work, we train a team of few robots learning from the sequenced steps taken by multiple people to complete a task that has many variations. Once trained via DL, the robot system is able to complete a variant of the task. We demonstrate this in a toy building construction, using blocks, two mobile robots, and a fixed robot arm using DL. There is a camera with a field of view covering the whole workspace. This enables the location information of the blocks and the robots to be available to the system. We do the training by either getting the people to do variants of tasks using the AR system implemented on a tablet computer or using a simulator that mimics the actual system. This results in a system that (deep) learns through AR or simulator based user interaction to achieve the completion of a previously unseen, relatively complex variant of a task. We are not concerned about the optimal way of doing a task in this work, but we just borrow the knowledge that has been mastered by humans over a long period of time.

The contributions of our work includes

1) Capturing human interactions with the robot system, particularly, using augmented reality or a simulator with the objective of learning a relatively complex task

to be performed by a team of robots,
2) The use of a sequence of occupancy grids to encode the sequence of actions base and
3) The deep learning method, particularly for a sequential task.

To the best of our knowledge, this is the first t ime that a robot system learns from multiple humans performing variants of a task to complete a previously unseen variant of a task.

The rest of the paper is organized as follows: section II presents the literature review; section III and IV cover the method, where section IV focuses on the deep learning method; section V presents the results followed by the conclusion in section VI.

II. Related Work

In this section, related work is detailed according to the four (4) main sub-systems (detailed in section III) of the proposed system. Nevertheless, none of the work have amalgamated all of them together to transfer human-like behavior to a robotic team.

A. Learning Frameworks

The problem that we are going to solve can be identified as a sequential decision-making problem, where the system needs to predict the actions needed to get to the final goal while trying to "act as human" as possible.

Learning policies for decision-making problems through neural networks are addressed, especially in gaming environments in [5]. The approach taken by Mnih *et al*. in [5] builds a dataset of previous experience, similar to the dataset we build, using batch RL to train large convolutional neural networks in a supervised learning setting.

Imitation learning is used to transfer human knowledge by matching the performance of the demonstrator. One popular algorithm, DAGGER [6], iteratively produces new policies based on polling the expert policy outside its original state space and our proposed algorithm has certain similarities to it. However, DAGGER needs the demonstrator to be available during training to provide feedback to the agent. Duan *et al*. [7] uses one-shot imitation learning where the agent is provided with an entire demonstration as input in addition to the current state. The problem with [7] is that it requires a distribution of tasks with different initial conditions and goal states, and the agent can never learn to improve upon the demonstrations.

Use of Apprenticeship Learning (AL) and Reinforcement Learning (RL) for knowledge transfer and policy formation can be found in [8]. However, it was applied for modeling helicopter dynamics and deriving control policies, not to automation task of a previously unseen variant of a task using many robots.

Also, there has been interest in combining imitation learning and RL. For example, the algorithm by Taylor *et al*. [9] transfers knowledge directly from human policies. Brys *et al*. [10], Suay *et al*. [11] have shown how expert advice or demonstrations can be used to shape rewards in the RL problem.

AlphaGo [12] uses a demonstration data set consisting of expert actions to train a policy network using supervised learning before interacting with the real task. It uses this learning to apply policy gradient updates during self-play, combined with planning roll-outs. AlphaGo is game centric where we are more focused on bridging the gap between human and robot behaviours because we are not focused on the optimality of the system.

Deep Q-learning from Demonstrations (DQfD) [13] uses pre-training to learn to imitate the demonstrator with a value function that satisfies the Bellman equation so that it can be updated with temporal differences updates once the agent starts interacting with the environment. DQfD has only been implemented in gaming environments and have used millions of iterations to train the models.

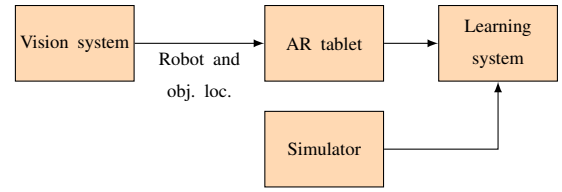
B. Multi-robot Teams

The system described in Frank *et al*. [14] has an augmented reality-based platform for object manipulation using a stationary robot arm. It includes a robotic platform with two 6 degree-of freedom (DoF) arms, a table with coloured blocks of different shapes, and a tablet device held by the user. The robotic system mentioned in Frank *et al*. [15], consists of many wheeled mobile robots. In his system, mobile robots electronics include a motor controller that drives the wheels and gripper, as well as a single board computer that provides Wi-Fi connectivity and allows robots to subscribe to commands published by a tablet computer.

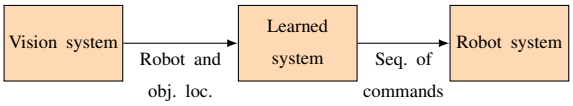
Motion planning methods by Bruce *et al*. [16], control methods by Chen *et al*. [17] of multiple mobile robots, and object transportation using multiple robots are discussed in many research works [18]. Furthermore, occupancy grids are used for robot navigation in Elfes *et al*. [19]. However they are not used for structure representation in the context of a learning task. Fiala *et al*. [20] has used overhead cameras to track the robots using fiducial markers and to control them.

C. Augmented Reality Based Robot Control

Frank *et al*. [15] has developed a novel mobile mixed-reality approach that allows operators to interact with and control systems of mobile robots, endowed with fiducial markers, without the need for a structured environment or specialized, research-grade equipment. In his paper, he discusses two approaches of mapping user interactions to robot commands as the device is moved arbitrarily by the operator, by estimation of a transformation between a coordinate frame attached to the device and a fixed reference frame. The workspace used for above study proposed mobile mixed-reality interfaces using open-source third-party libraries for computer vision, graphics rendering, and communication over the robots network. Frank *et al*. [14] [21] also discusses about controlling robots including a robot arm using an augmented reality-based application using fiducial markers and mapping functions. Those methods are directly used in our application.



(a) Training phase



(b) Testing phase

Figure1. System overview.

III. Methodology

First, the users are advised to sketch a toy building to their liking either using the developed simulator platform or the AR system implemented on a tablet computer. Next, the users will construct the building they sketched by moving two types of blocks (cubes and cylinders), and these sequences of actions taken by the users will be recorded. At the same time, the actions are materialized in the physical world with the help of a robot team to actually build the toy building. There is an overhead camera, which keeps track of the blocks and the robots using fiducial markers, to help the building constructing process. The recorded sequence of actions together with the building structure are deep learned by the system so that the system becomes capable of completing a previously unseen, relatively complex variant of a building task.

The proposed system consists of 4 sub-systems to achieve the objectives.

- Deep learning environment to learn and predict the sequence of actions
- Hardware system including the robots and the workspace.
- Augmented reality based robot control for the learning model.
- The vision system with ALVAR markers. [22]

The main focus is on the deep learning system that learns from humans to make the team of robots achieve the task. The robots and the vision system, which we discuss up to a certain extent, are used for proving the concept.

Figure 1 shows the system overview in the training phase and testing phase. In the training phase, the learning system receives the sequences of action taken by multiple humans in achieving a specified variant of the task. This can be either by moving the actual robots using the augmented reality tablet in its manual mode or using the simulator environment. In the testing phase, once the the previously unseen variant of the task is given, the learned system issues a sequence of commands to the robot system (in the automatic mode) to achieve the task. Figure 2 shows a picture of the proof-of-concept system implementation.

A. Workspace

The workspace consists of three main areas. The material placement area is used to place blocks required for the construction. The blocks used in building the structure consist of two different types (i.e. cubes and cylinders) that are placed randomly in the workspace. Two mobile robots carry these blocks to the unloading area. Finally, the robot arm manipulates these blocks to construct the toy building. Each level in the construction area comprises of 3×5 locations, and there are 3 levels, giving rise to 45 variants of locations. The overall workspace is shown in Figure 3.

B. The Vision System and ALVAR Markers

We have used an overhead camera feed to obtain the locations of the robots and the construction blocks using fiducial marks attached to them.

We have used Robot Operating System (ROS) [23] as the main software platform to control the system. ALVAR markers [22] are used as the fiducial markers, which have unique patterns that can be detected by its ROS library. Finally, using the marker tracking information we create the map of the workspace, and it is visualized using ROS Visualization (Rviz) tool.

C. Robot System

The robot system comprises of two mobile robots and one robot arm, that functions collaboratively to achieve a final goal of constructing a given building pattern on the workspace using different types of blocks. The dimensions of a mobile robot is $30\,\text{cm} \times 20\,\text{cm} \times 9\,\text{cm}$ without the arm expansion. We have placed a $12\,\text{cm} \times 12\,\text{cm}$ sized ALVAR marker on top of the robot to uniquely identify each mobile robot. The mobile robot arm has a simple gripping mechanism, where two grippers are controlled using a servo motor. Figure 4(a) shows one of the actual mobile robots designed for our system.

A ROS node [23] written in Python is capable of moving the blocks from an initial position of the block to the unloading area as commanded. In the manual mode, this node directly listens to the commands coming from a node running on a tablet computer. In the automatic mode, commands generated from the DL model reach this node, through a controller node running on the server computer.

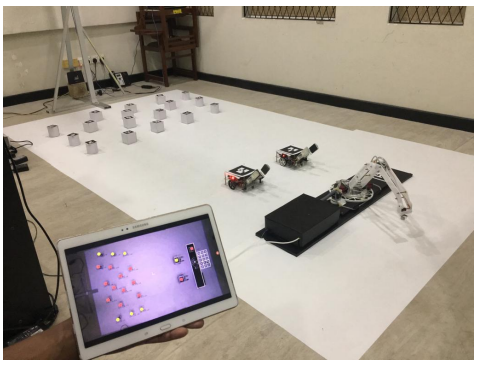


Figure 2. The proof-of-concept system implementation.

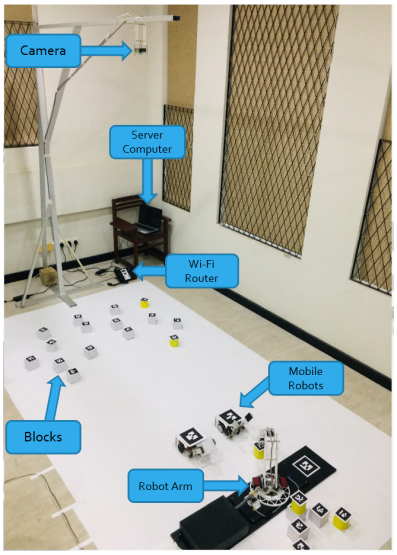
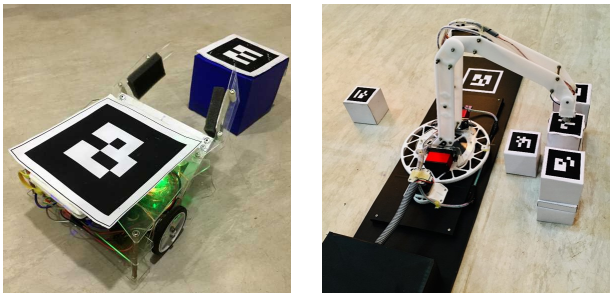


Figure 3. The workspace with robots, and objects.

When the robot is given a final goal, it analyzes this dynamic map and accordingly moves towards the goal while avoiding the obstacles on its path. In our robot navigation algorithm, few points are simulated ahead of the robot path and the point which is closest to the goal and guarantees no collision with blocks is chosen as a sub-goal. Then, the sub-goals are iteratively defined as the robot move towards the goal as mentioned above. The above process is repeated until the robot reaches the actual goal with a tolerance distance.

The robot arm designed for our system has 5 Degrees-of-Freedom (DoF). Its first three joints are used to position the end effector and the last two joints are used to adjust the orientation of the end effector. We used four 180-degree servo motors: two FEETECH Ultra-High-Torque Digital Giant Servos (FT5335M), one Pololu Power HD High-Torque Servo (1501MG) and one TowerPro MG90 - Micro Servo. We have used a NEMA 17 stepper motor at the base of the robot arm and a gear drive with a gear ratio of 1:4 in the mechanical design, which allows for horizontal circular motion of the whole structure. We have used an electromagnetic mechanism to pick up the blocks as they are of different shapes.

The actual design of the robot arm is shown in the Figure 4(b). The server computer can send the position of blocks



(a) Mobile robot (b) Robot arm

Figure 4. Actual robots.

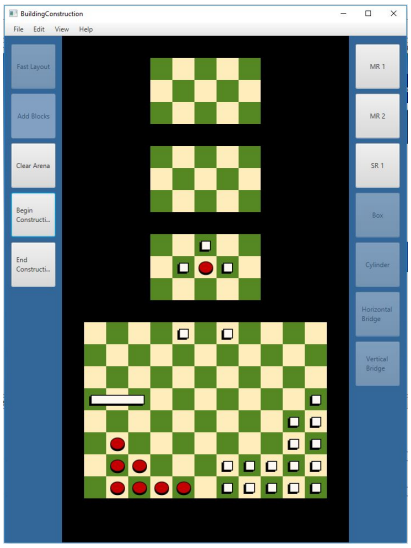


Figure 5. GUI of computer application.

with respect to the fixed base of the arm by analyzing the markers on the blocks as well as arm. Our inverse kinematics model [24] outputs the joint angles $(\theta_1, \theta_2, \theta_3, \theta_4)$. Servo angles calculated using the joint angles, are input to the servo controller.

D. Data Collection Interfaces

We have developed two mechanisms to collect data for our conceptual system.

1) A GUI based computer application using JavaFX, where the user can choose one of the initial layouts for the workspace or make his own initial layout and move blocks around very easily using the mouse pointer—The workspace is represented as a grid where each cell represents a state: occupied, not-occupied, and occupied with specific type of block. The state of each grid cell (time evolution) is tracked through the application when the user gives inputs to change the block positions in the grid. This data is later re-read in Python for the model training. Figure 5 shows the user interface of the computer application for data collection.
2) A mobile application used to control the mobile robots in AR environment, for those who like to interact with robots—The user can command the robots directly through this application and we will capture the users sequence of actions in order to train our system. Figure 6(a) shows the user interface of the manual control mode of the system.

E. Android Application for Controlling the System

We have developed an Android application, which works as a ROS node that can send and receive messages to all the nodes connected to the same master ROS node. Figure 6(b) shows the user interface of the autonomous control mode of the system.

The autonomous control mode is used to give the final building structure to the system by the user. When the user

sends the building structure, the system predicts the sequence of control actions and sends that information to the hardware system. Next, it performs this sequence of actions and builds the final building structure which is comparable to a man-made building.

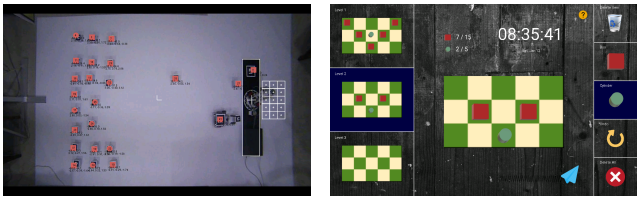
IV. Machine Learning Model

Our machine learning system predicts the most probable move (block type and the destination) depending on the current state defined by a 4-D occupancy grid. A training pattern is an action taken by a human being, given the current occupancy grid. Here, the process can be thought of as a Markov process and can be represented using Markov chains where there is a probability of transitioning from one state to another. We use a DNN to calculate these state transitioning probabilities. This system, after training, enables us to get the robot system to sequentially move blocks to ultimately achieve the specified variant of the task—building a specified building in our proof-of-concept example.

In this architecture, we represent the current state using a 4-D occupancy grid representation using binary values. Each cell that needs to be filled is represented using one hot encoding, and the cells which are already occupied—because of the previous interactions—are represented as negative one hot encoding. The grid is 4-D to account for the three levels of the building in our proof-of-concept example and to handle a type of a block (e.g., cube, cylinder) per physical occupancy cell. [25] Given the occupancy grid, the user has the option of selecting one out of 135 (5 spatial, 3 spacial, 3 vertical, 3 types of blocks) actions. The simulator or the actual AR user input system saves the training data in the aforementioned occupancy grid format used subsequently for training along with the action.

Here we draw inspiration from the Deep Q Learning (DQN) [13] networks in designing the network. In DQN, a DNN is used to approximate the q value function $Q^*(s,a)$, where each action a is given a q value depending on the sate s. In policy gradient methods [26], neural networks are instead used to represent a policy $\pi_\theta(\mathrm{a}|\mathrm{s})$. In our work, the policy is stochastic, i.e., it provides a distribution over the possible set of actions. The policy is learnt by the user interactions. We use a similar procedure to learn the sequence of actions needed to complete a given structure.

We compute the posterior probability of the uncertain proposition, (the way a human would have chosen to build a given structure) using DNN. Here, the posterior probability is the probability distribution of an unknown occupancy grid, treated as a random variable, conditional on the evidence obtained from the sequences of actions taken by humans. DNN, a 3D CNN, is able to give a posterior distribution of the set of actions of a given structure.



(a) Manual mode (b) Autonomous mode

Figure 6. User interfaces of the system.

Figure 7 shows the architecture of the DNN model used. We have used categorical crossentropy J as loss function. The model predicts 1 out of 135 actions, where an action is defined by the cell location and the block types.

Training data does not cover all the state-action pairs. Hence, the model does not have access to all the transition probabilities, as training data are insufficient to cover probability distributions over all the building structures. Here we ask the question whether it possible to get the probabilities for the complete set of state action pairs and achieve the task, given the final state to mimic the set of actions a human would take.

We have used two algorithms for the ML model. The algorithm 1 is used to pre-process data. This generates a sequence of state action pairs D^{replay} from training data. After that we use the algorithm 2 to train the DNN to be able to predict the next action to be taken while using P not collected from humans.

Algorithm 1 Pre-processing: state action pair D^{replay} creation

Require: $\{(B,A)\}$: building structure B (variant of a task), and sequence of actions $A = \{a_1, a_2, \ldots, a_n\}$.
Ensure: D^{replay} state, action tuples $\{(s,a)\}$

```
for each item in {B,A} do
    s = B
    for each action a in A do
        Add (s,a) to D^replay.
        Update state s, replace occupancy grid cell value
        w.r.t. the action a
    end for
end for
```

V. Experiments and Results

In order to establish that our robot team can carry out an unseen variant of a task after learning from multiple trials carried out by humans, we carried out three experiments: 1) Investigating whether humans stick to pattern given a variant of a task, 2) Getting the robots to build unseen variants of building (structures) after by learning, 3) Investigating the effect of using top-5 predictions of the DNN.

1) The main aim of this study was to build a ML friendly environment to capture the user interactions in a sequential task. For the selected example, it is the sequence of block placements carried by humans. Humans usually approach this sequential task by following a pattern if they identify one at the outset. In order to verify this, we gave 50 people 11 variant of tasks and asked them to fill the sequence of placement using numbers. Figure 8 shows few of variant of tasks given to humans. First three structures have clear patterns while the final structure has a random

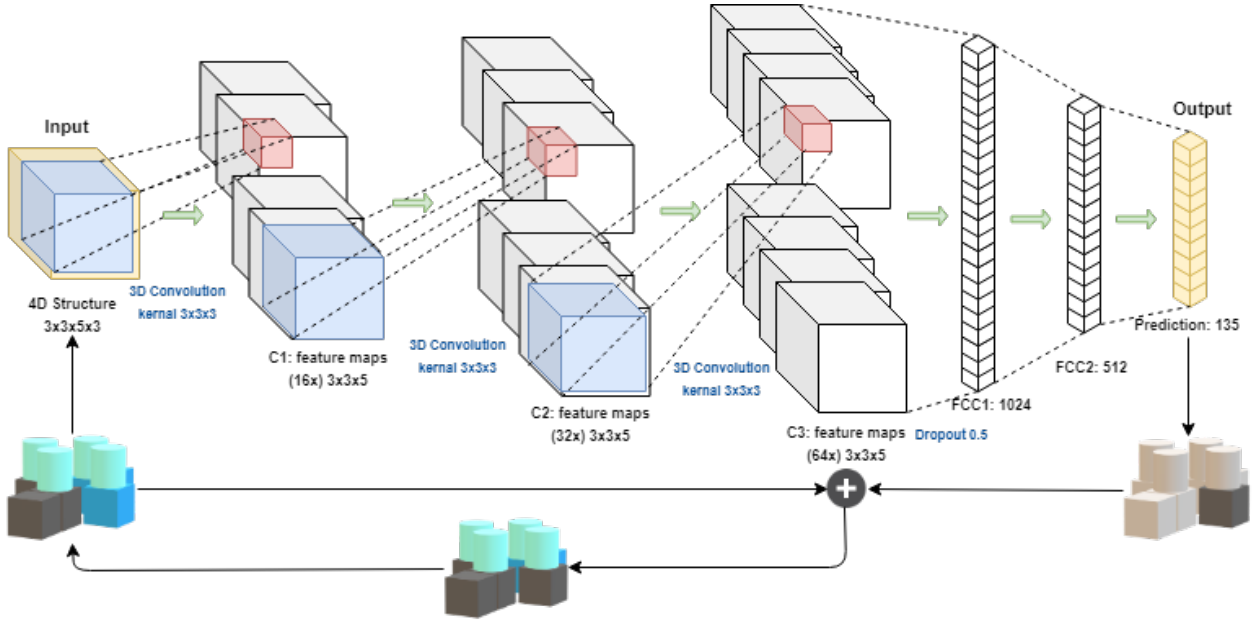


Figure 7. 3-D CNN architecture of the model.

Algorithm 2 Deep learning from D^{replay}

Require: D^{replay}: state-action pairs (Algorithm 1).
Require: θ: weights initialization for the DNN.
Require: k: number of epochs.
Require: P: patterns not included in D^{replay}.
Require: k: number of exploratory gradient updates.
Ensure: θ: updated weights for the DNN.

```
Step 1:
for steps t ∈ {1,2,...,k} do
    Sample a mini-batch of n transitions from D^replay.
    Calculate loss J using CNN network.
    Perform an epoch to update θ.
end for
Step 2:
for steps t ∈ {1,2,...,k'} do
    for each pattern p in P do
        Calculate action a using the network for s = p.
        Sample action from 3 highest probable actions a
        and validate the action
        If a is erroneous: break
        Add (s,a) to D^replay.
        Update s
        Iterate until completion of the pattern p
    end for
    Do Step 1
end for
```

block placement. There are three separate occupancy grids for three levels of the building.

We observed that for structure 1, 94% of the building patterns were similar. For structures 2 and 3, the corresponding values were 82% and 86%, respectively. . On the other hand, if there is no pattern visible (as in structure 4), they seemed to resort to seemingly a random sequences. This confirms that

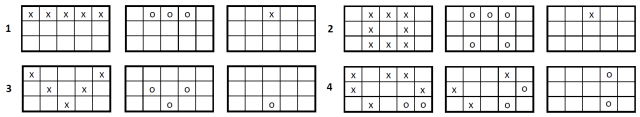


Fig. 8: Few structures (variants of tasks) given. Boxes are marked with "x" and cylinder are marked with "o".

humans, in general, adhere to a pattern in taking a sequence of actions.

We have given the same 11 structures to our ML model and the model was able to predict all the moves correctly. Figure 9 shows the ML model output of few them. These patterns are identified correctly by the system and the action sequences are predicted similar to how a human would have preferred to build that structure.

2) We trained the system using 300 different building structures. 50 previously unseen structures were given and the below results were analyzed considering only the highest probable action selected by the model. For 34 buildings out of 50, a correct action sequence was 100% accurately predicted by the system. 45 out of 50 buildings had more than 75% accuracy. 523 out of 590 states were correctly predicted by the model. Therefore, the average accuracy of the system is 88.64%. Most errors were due to the inability to identify the correct block type when there was no identifiable pattern in the building structure.

3) Additionally, we have tested the accuracy of the system, considering the input state s and predicted action a with the actions taken by the demonstrators. We have considered the 5 most probable actions predicted by the system for the results comparison. The results obtained from the above experiment are shown in the TABLE I. Accuracy columns in TABLE I represent accuracy values considering the training data set and the testing data set drawn from D^{replay} in algorithm (1).

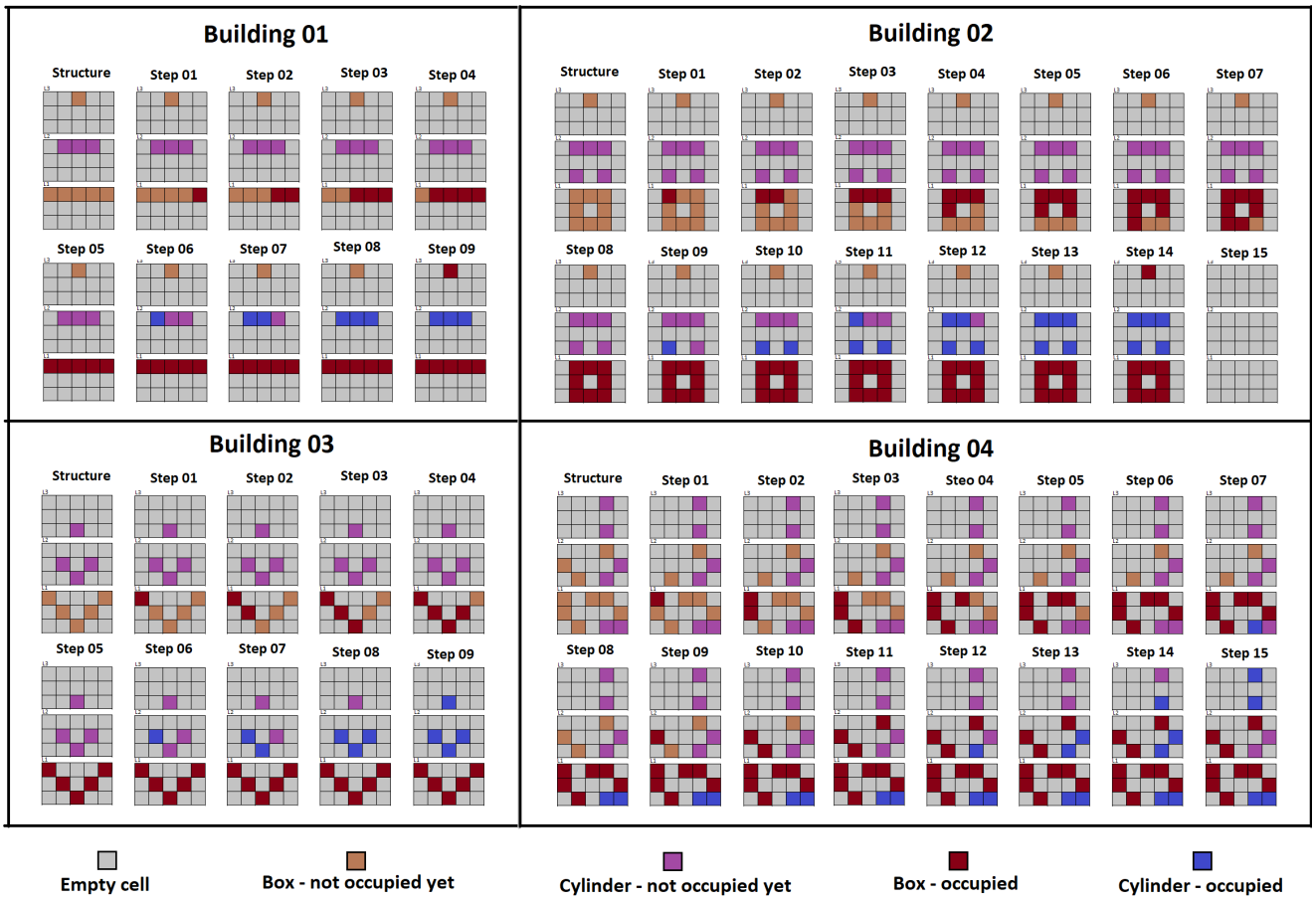


Figure 9. Predicted building structures by the model. Top-left in each box: specified structure. Other boxes: states visited denoting the sequence of predictions. Top-left and bottom-right box in each cell are identical indicating the correctness of the predictions of the DNN. Best seen in color.

TABLE I: ACCURACY OF THE ML MODEL

	Training Accuracy	Testing Accuracy
First choice only	88.64%	75.60%
First 2 choices	92.90%	83.10%
First 3 choices	95.30%	87.40%
First 4 choices	97.50%	90.40%
First 5 choices	98.60%	92.10%

Results conclude that the system is able to capture human behaviour and perform the sequence of actions in a human-like manner.

VI. CONCLUSION

In this work, we developed a system for training a set of robots to achieve a relatively complex task. We collected data to train a deep convolutional network by asking humans to carry out a variant of the task either using the AR platform or simulator. We recorded the gathered training data in the form of a sequence of occupancy grids and actions taken. Then we trained a deep convolutional network. In the testing time, this network predicts the next action based on the current occupancy grid. Such a sequence of actions leads to the robot team completing a previously unseen variant of the task. We achieved more than 80% accuracy in taking the top-predicted action. We achieved even better accuracy by choosing one of top five actions, as we have a notion of the validity of a predicted action.

In future, we need to investigate incorporating constraints to ban predication of invalid actions, increasing the size of the robot team, and increasing the complexity of the task.

REFERENCES

[1] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, "Neural combinatorial optimization with reinforcement learning," *arXiv preprint arXiv:1611.09940*, 2016.

[2] S. Calinon, F. Dhalluin, E. Sauser, D. Caldwell, and A. Billard, "A probabilistic approach based on dynamical syst. to learn and reproduce gestures by imitation," *IEEE Robot. Autom. Mag.*, vol. 17, no. 2, pp. 44–54, 2010.

[3] M. Laskey, C. Chuck, J. Lee, J. Mahler, S. Krishnan, K. Jamieson, A. Dragan, and K. Goldberg, "Comparing human-centric and robot-centric sampling for robot deep learning from demonstrations," in *Proc. IEEE Int. Conf. Robot. Autom.*, Singapore, May 2017, pp. 358–365.

[4] N. Michael, M. M. Zavlanos, V. Kumar, and G. J. Pappas, "Distributed multi-robot task assignment and formation control," in *Proc. IEEE Int. Conf. Robot. Autom.*, Pasadena, CA, May 2008, pp. 128–133.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[6] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proc. of the fourteenth Int. Conf. on artificial intelligence and statistics*, 2011, pp. 627–635.

[7] Y. Duan, M. Andrychowicz, B. Stadie, O. J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba, "One-shot imitation learning," in *Proc. Adv. in Neural Inf. Process. Syst.*, 2017, pp. 1087–1098.

[8] P. Abbeel and A. Y. Ng, "Exploration and apprenticeship learning in reinforcement learning," in *Proc. Int. Conf. Mach. Learning*. ACM, 2005, pp. 1–8.

[9] M. E. Taylor, H. B. Suay, and S. Chernova, "Integrating reinforcement learning with human demonstrations of varying ability," in *Proc. Int. Conf. on Auton. Agents and Multiagent Syst.*, 2011, pp. 617–624.

[10] T. Brys, A. Harutyunyan, H. B. Suay, S. Chernova, M. E. Taylor, and A. Nowé, "Reinforcement learning from demonstration through shaping," in *Twenty-Fourth Int. Joint Conf. on Artificial Intelligence*, 2015.

[11] H. B. Suay, T. Brys, M. E. Taylor, and S. Chernova, "Learning from demonstration for shaping through inverse reinforcement learning," in *Proceedings of the 2016 Int. Conf. on Autonomous Agents & Multiagent Syst.* Int. Foundation for Autonomous Agents and Multiagent Syst., 2016, pp. 429–437.

[12] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.

[13] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, G. Dulac-Arnold, J. Agapiou, J. Z. Leibo, and A. Gruslys, "Deep q-learning from demonstrations," in *AAAI Conf. on Artificial Intell.*, New Orleans, LA, 2018.

[14] J. A. Frank, M. Moorhead, and V. Kapila, "Realizing mixed-reality environments with tablets for intuitive human-robot collaboration for object manipulation tasks," in *IEEE Int. Symp. on Robot and Human Interactive Commun.*, New York, NY, 2016, pp. 302–307.

[15] J. A. Frank, S. P. Krishnamoorthy, and V. Kapila, "Toward mobile mixed-reality interaction with multi-robot syst." *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 1901–1908, 2017.

[16] J. Bruce and M. Veloso, "Real-time randomized path planning for robot navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot and Sys.*, vol. 3, Lausanne, Switzerland, 2002, pp. 2383–2388.

[17] Q. Chen and J. Luh, "Coordination and control of a group of small mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.* IEEE, 1994, pp. 2315–2320.

[18] A. Yamashita, T. Arai, J. Ota, and H. Asama, "Motion planning of multiple mobile robots for cooperative manipulation and transportation," *IEEE Trans. Robot. Autom.*, vol. 19, no. 2, pp. 223–237, 2003.

[19] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[20] M. Fiala, "A robot control and augmented reality interface for multiple robots," in *Canadian Conf. on Comput. and Robot Vision*. IEEE, 2009, pp. 31–36.

[21] J. A. Frank, M. Moorhead, and V. Kapila, "Mobile mixed-reality interfaces that enhance human–robot interaction in shared spaces," *Frontiers in Robot. and AI*, vol. 4, p. 20, 2017.

[22] C. Woodward, J. Lahti, J. Rönkkö, P. Honkamaa, M. Hakkarainen, J. Jäppinen, K. Rainio, S. Siltanen, and J. Hyväkkä, "Case digitalo-a range of virtual and augmented reality solutions in construction application," in *24th W78 Conf., Maribor. In: 24th W*, vol. 78, 2007, pp. 529–540.

[23] "Robot Operating System (ROS) - Documentation Guide." [Online]. Available: http://wiki.ros.org/

[24] M. W. Spong, S. Hutchinson, M. Vidyasagar *et al.*, *Robot modeling and control*, 2006, ch. Forward and inverse kinematics.

[25] D. Meyer-Delius, M. Beinhofer, and W. Burgard, "Occupancy grid models for robot mapping in changing environments," in *AAAI Conf. on Artificial Intell.*, Toronto, Ontario, Canada, 2012.

[26] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learning*, Beijing, China, 2014, pp. 387–395.