Robotics: Science and Systems 2019
Freiburg im Breisgau, June 22-26, 2019

DIViS: Domain Invariant Visual Servoing for Collision-Free Goal Reaching

Fereshteh Sadeghi
University of Washington

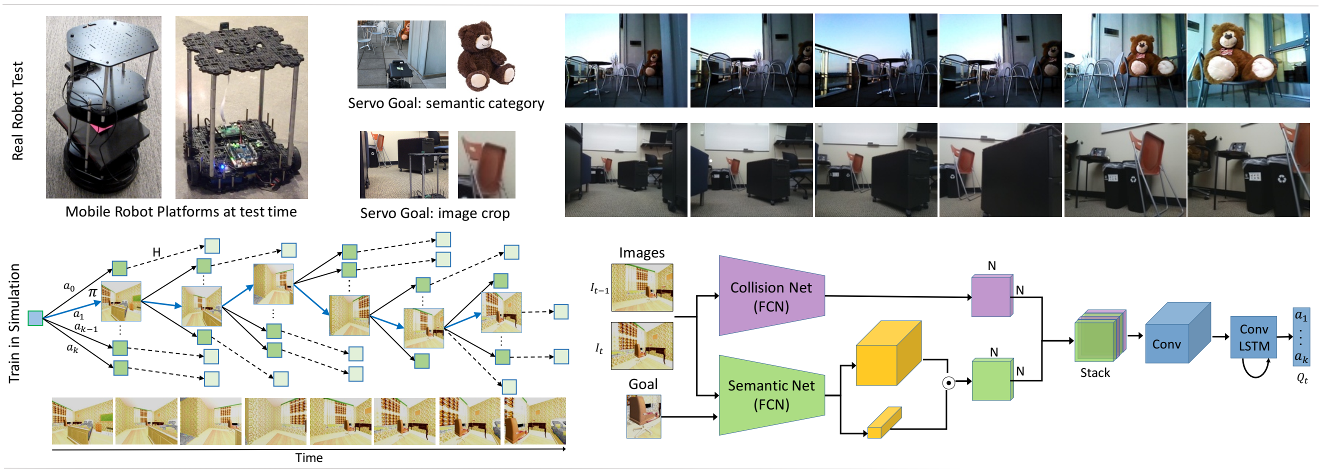

Fig. 1: Domain Invariant Visual Servoing (DIVIS) learns collision-free goal reaching entirely in simulation using dense multi-step rollouts and a recurrent fully convolutional neural network (bottom). DIViS can be directly deployed on real physical robots with RGB cameras for servoing to visually indicated goals as well as servoing to semantic object categories (top).

***Abstract*—Robots should understand both semantics and physics to be functional in the real world. While robot platforms provide means for interacting with the physical world they cannot autonomously acquire object-level semantics without needing human. In this paper, we investigate how to minimize human effort and intervention to teach robots perform real world tasks that incorporate semantics. We study this question in the context of visual servoing of mobile robots and propose DIViS, a *D*omain *I*nvariant policy learning approach for collision free *Vi*sual *S*ervoing. DIViS incorporates high level semantics from previously collected static human-labeled datasets and learns collision free servoing entirely in simulation and without any real robot data. However, DIViS can directly be deployed on a real robot and is capable of servoing to the user-specified object categories while avoiding collisions in the real world. DIViS is not constrained to be queried by the final view of goal but rather is robust to servo to image goals taken from initial robot view with high occlusions without this impairing its ability to maintain a collision free path. We show the generalization capability of DIViS on real mobile robots in more than 90 real world test scenarios with various unseen object goals in unstructured environments. DIViS is compared to prior approaches via real world experiments and rigorous tests in simulation. For supplementary videos, see: https://fsadeghi.github.io/DIViS**

I. Introduction

Perception and mobility are the key capabilities that enable animals and human to perform complex tasks such as reaching food and escaping predators. Such scenarios require the intelligent agent to make high level inference about the physical affordances of the environment as well as its semantics to recognize goals and distinguish walkable areas from obstacles to take efficient actions towards the goal. Visual servoing is a classic robotic technique that relies on visual feedback to control the motion of a robot to a desired goal which is visually specified [24, 12, 11, 65]. Historically, visual servoing has been incorporated for both robotic manipulation and navigation [24, 12, 4, 10, 27]. Figure 2 depicts an early visual servoing mobile robot of 1995 that servos to a goal image by matching geometric image features between the view at the final desired position and robot's camera view [47, 15]. While visual servoing mechanisms aim to acquire the capability to surf in the 3D world, they inherently do not incorporate object semantics. In addition, conventional servoing mechanisms need to have access to the robot's view in the final goal position. Such requirement can restrict applicability as it may be infeasible to have the camera view at the goal position specially in unstructured unknown environments.

Deep learning has achieved impressive results on a range of supervised semantic recognition problems in vision [30, 18] language [38] and speech [23, 20] where supervision typically comes from human-provided annotations. In contrast, deep reinforcement learning (RL) [41, 53, 40] have focused on learning from experience, which enables performance of physical tasks, but typically do not focus on widespread semantic generalization capability needed for performing tasks in unstructured and previously unseen environments. Robots, should be capable of understanding both semantics and physics in order to perform real world tasks. Robots have potential to autonomously learn how to interact with the physical world.

However, they need human guidance for understanding object-level semantics (e.g., chair, teddy bear, etc.). While it is exceedingly time-consuming to ask human to provide enough semantic labels for the robot-collected data to enable semantic generalization, static computer vision datasets like MS-COCO [35] or ImageNet [30] already offer this information, but in a different context. In this work, we address the question of how to combine autonomously collected robot experience with the semantic knowledge in static datasets, to produce a semantic aware robotic system.

We study this question in the context of goal reaching robotic mobility in confined and cluttered real-world environments and introduce DIViS, **D**omain **I**nvariant **Vi**sual **S**ervoing that can maintain a collision-free path while servoing to a goal location specified by an image from the initial robot position or a semantic object category. This is in contrast to the previous visual servoing approaches where the goal image is taken in the final goal position of the robot. Performing this task reflects robots capability in understanding both semantics and physics; The robot must be able to have sufficient physical understanding of the world to reach objects in cluttered rooms without colliding with obstacles. Also, semantic understanding is required to associate images of goal objects, which may be in a different context or setting, with objects in the test environment. In our setup, collisions terminate the episode and avoiding obstacles may necessitate turning away from the object of interest. This requires the robot to maintain memory of past movements and learn a degree of viewpoint invariance, as the goal object might appear different during traversal than it did at the beginning of the episode. This enforces the robot to acquire a kind of internal model of "object permanence" [3, 7, 8]. Similar to how infants must learn that objects continue to exist in their previous position once they are occluded, the robot must also learn that an object specified by the user continues to exist.

The main contribution of our work is a novel domain invariant policy learning approach for direct simulation to real world transfer of visual servoing skills that involve object-level semantics and 3D world physics such as collision avoidance. We decouple physics and semantics by transferring physics from simulation and leveraging human annotated real static image datasets. Therefore, our approach addresses the challenges and infeasibilities in collecting large quantities of semantically rich and diverse robot data. To keep track of changes in viewpoint, our visual servoing model maintains a short-term memory via a recurrent architecture. In addition, our method predicts potential travel directions to avoid colliding with obstacles by incorporating a simple reinforcement learning method based on multi-step Monte Carlo policy evaluation. We conduct wide range of real-world quantitative and qualitative evaluations with two different real robot platforms as well as detailed analysis in simulation. Our policy which is entirely trained in simulation can successfully servo a real physical robot to find diverse object instances from different semantic categories and in a variety of confined and highly unstructured real environments such as offices and homes.

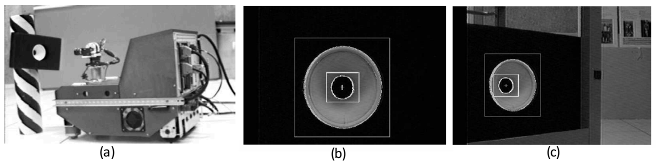

Fig. 2: (a) The classic 1995 visual servoing robot [47, 15]. The image at final position (b) was given as the goal and the robot was started from an initial view of (c).

II. Related Work

Visual servoing techniques rely on carefully designed visual features and may or may not require calibrated cameras [12, 15, 62, 65, 27]. Our method does not require calibrated cameras and does not rely on geometric shape cues. While photometric image-based visual servoing [10] aims to match a target image, our approach can servo to goals images which are under occlusion or partial view. Recently several learning based visual servoing approaches using deep RL are proposed for manipulation [34, 31, 51], navigation via a goal image [44, 69] and tracking [33]. In contrast to the goal-conditioned navigation methods of [44, 69], our approach uses the image of goal object from the initial view of the robot which can be partially viewed or heavily occluded. Also, our method learns a domain invariant policy that can be transferred to the real world despite the fact that it is entirely trained in simulation.

Transferring from simulation to real and bridging the reality gap has been an important area of research for a long time in robotics. Several early works include using low-dimensional state representations [42, 28, 13, 58]. Given the flexibility and diversity provided by the simulation environments, learning policies in simulation and transferring to the real world has recently gained considerable interest in vision-based robotic learning [50, 51, 48, 37, 29, 5]. Prior works on representation learning either learn transformation from one domain to another [19, 49] or train domain invariant representations for transfer learning [36, 6, 17]. [67] proposed inverse mapping from real images to renderings using CycleGAN [68] and evaluated on a self-driving application and an indoor scenario. Domain randomization for vision-based policy learning was first introduced in [50] for learning generalizable models that can be directly deployed on a real robot and in the real world. It was then broadly used for various robot learning tasks [51, 29, 61, 2, 46, 60, 45, 43, 56, 37]. In contrast, our focus is on combining transfer from simulation (for physical world understanding) with transfer from semantically labeled data (for semantic understanding). The physical challenges we consider include navigation and collision avoidance, while the semantic challenges include generalization across object instances and invariance to nuisance factors such as background and viewpoint. To our knowledge, direct simulation to real world transfer for semantic object reaching with diverse real-world goals and environments is not explored before.

Visual navigation is a closely related problem to our work. A number of prior works explored navigation via deep RL using recurrent policy, auxiliary tasks and mixture of multiple objectives in video game environments with symbolic targets that do not necessarily have the statistics of real indoor scenes [40, 14, 39, 26]. We consider a problem setup with realistic rooms populated with furniture and our goal is defined as reaching specific objects in realistic arrangements. Vision-based indoor navigation in simulation under grid-world assumption is considered in several prior works [69, 21]. [66] addressed visual navigation via deep RL in real maze-shaped environment and colored cube goals without possessing semantics of realistic indoor scenes. [52] benchmarked basic RL strategies [14, 39, 40]. With the increasing interest in learning embodied perception for task planning and visual question answering, new simulated indoor environments have been developed [52, 63, 55, 1, 16]. While these environments facilitate policy learning in the context of embodied and active perception, they do not consider generalization to the real world with a physical robot. As opposed to all aforementioned prior works, our work focuses on transfer of both semantic and physical information, with the aim of enabling navigation in real-world environments. We do not focus on long-term navigation (e.g., between rooms) or textual navigation [1], but we focus on local challenges of highly confined spaces, collision avoidance, and searching for occluded objects which require joint understanding of physics and semantics.

Simultaneous localization and mapping (SLAM) has been used for localization, mapping and path planning. One group of SLAM methods creates a pipeline of different methods to first build a map using geometry constraints and then performs planning in the estimated map to navigate. Another group traverses to build representation of a new environment and simultaneously plans to navigate to a goal. While these approaches provide promising solutions for building the map and navigation, their main focus is not on visual goal reaching and transfer. There is a large body of work on SLAM for which we refer the reader to these comprehensive surveys [59, 9].

III. Diverse Collision Free Goal Reaching

We build a new simulator for the collision-free visual goal-reaching task using a set of 3D room models with diverse layouts populated with diverse furniture placements. Our goal is to design diverse tasks that require the agent to possess both semantic and physical understanding of the world. We define various object-reaching navigation scenarios where in each scenario we specify a different visual goal and place the agent in a random location and with a random orientation. The agent must learn a single policy capable of generalizing to diverse setups rather than memorizing how to accomplish an specific scenario.

Figure 3 depicts examples of our tasks. In each example, the agent is tasked to get as close as possible to a visually indicated goal while avoiding collisions with various obstacles. The map of the environment is unknown for the agent and the agent is considered to be similar to a ground robot which can only move in the walkable areas. This scenario is aligned with real-world settings where we have diverse environments populated with variable furniture. The furniture and room objects are not registered in any map, and the robot should be able to move towards objects without collision, since otherwise it can

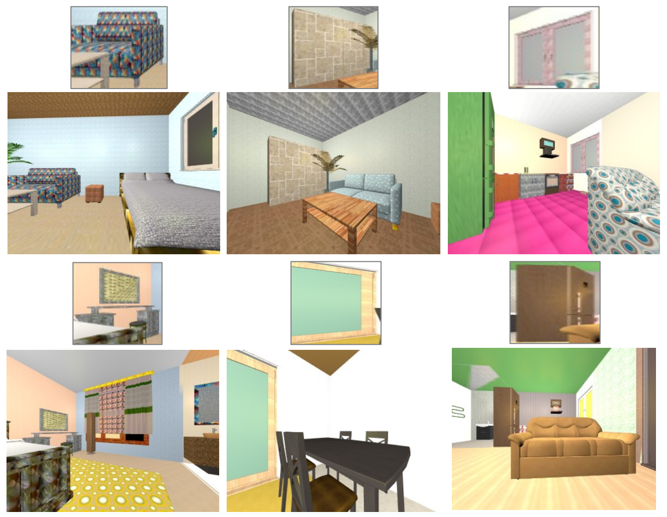
Fig. 3: Examples of the diverse goal-reaching mobility tasks. In each task, agent should reach a different visually indicated goal object. In each scenario, we use domain randomization for rendering.

damage itself and the environment. Collisions terminate the episode which also enforces the agent to learn an efficient policy without needing to often backtrack.

In our setup, the robot action is composed of rotation τ and velocity v. We consider a fixed velocity which implies that a fix distance is traversed in each step. We discretize the rotation range into $K-1$ bins and the action at each step is the rotation degree corresponding to the chosen bin. We also add an stop action making the total number of actions equal to K. Note that although our action space is discrete, we move the agent in a continuous space rather than a predefined discrete grid. Following the success of [50], we devise a randomized simulator where light and textures are randomized during rendering both at training and test time. In contrast to the prior work with grid world assumption and fixed rendering setup [69, 21], our setup creates a problem with *infinite* state space which is more realistic and more challenging.

IV. Domain Invariant Visual Servoing

The visual servoing task of collision-free goal-reaching involves multiple underlying challenges: (1) Learning to visually localize goal objects. (2) Learning to predict collision map from RGB images. (3) Learning the optimal policy π to reach the objects. To integrate rich visual semantics while learning the control policy we opt to disentangle perception from control. We first learn a semantically rich model $\Phi_{\hat{\theta}}$ to represent visual sensory input as observation o. Freezing the $\hat{\theta}$ parameters, we then learn the control policy π_{θ}.

A. Network Architecture

Our representation learning module consists of two fully convolutional networks both based on VGG16 architecture [54] which we call them as *Collision Net* and *Semantic Net*. Figure 1 shows our network architecture.

For Collision Net, we follow [50] to pre-train a free-space prediction network. Collision Net takes in an RGB image I and the output logit of its last convolutional layer provides

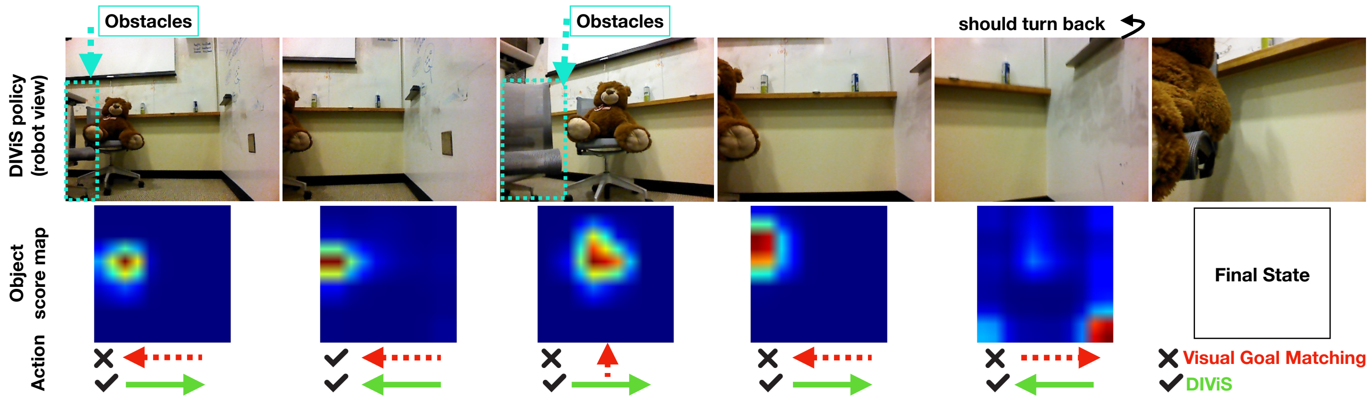

Fig. 4: Qualitative comparison of DIViS against Visual Goal Matching policy while the robot is tasked to reach "teddy bear". *Green arrows* show action directions taken by DIViS and *red arrows* (bottom row) show action directions chosen by Visual Goal Reaching which only relies on object score maps (middle row). Visual Goal Matching fails by colliding into obstacles while DIViS reaches the goal successfully by taking turns around the obstacles (top row).

an $N \times N$ collision map $\phi_{\hat{\theta}_c}(I)$ which predicts if an obstacle exists in the distance of 1 meter to the agent in its 2D ego-centric view. Figure 1 shows $\phi_{\hat{\theta}_c}(I)$ by a purple $N \times N$ map.

Our Semantic Net is built and trained based on [25]. We pre-train our Semantic Net on the MS-COCO object categories [35] with a weakly supervised object localization setup similar to [25]. We use the penultimate layer of the fully convolutional neural network of [25] to encode visual semantics in the spatial image space. Semantic Net describes each RGB image input via an $N \times N \times 2048$ map which we denote it as $S(I)$ and is shown by yellow box in Figure 1.

For each goal reaching task, our network takes in a goal image I^g that is fixed during the entire episode. At each timestep t, the network takes in the current image I_t, the previous image I_{t-1} as well as a goal image I^g. We use our Semantic Net to compute semantic feature representations $S(I^t)$, $S(I^{t-1})$ and $S(I^g)$, respectively. To represent the semantic correlation between the goal image and each of the input images in the 2D ego-centric view of the agent, we convolve the semantic visual representation of each input image with that of the goal image $\phi_{\hat{\theta}_s}(I, I_g) = S(I) \odot S(I^g)$. Also, for each pair of consecutive input images I_t and I_{t-1}, we compute the optical flow map $\psi(I_t, I_{t-1})$. We then stack the resulted pairs of collision maps $\phi_{\hat{\theta}_c}$, semantic correlation maps $\phi_{\hat{\theta}_s}$, and optical flow map ψ to generate the visual representation $\Phi_{\hat{\theta}}$ at each timestep t. For brevity we omit Is from the equation.

$$\Phi_{\hat{\theta}_t} = [\phi_{\hat{\theta}_c,t}, \phi_{\hat{\theta}_c,t-1}, \phi_{\hat{\theta}_s,t}, \phi_{\hat{\theta}_s,t-1}, \psi_{t,t-1}] \qquad (1)$$

At each timestep t, we feed the observation $o_t = \Phi_{\hat{\theta}_t}$ into our policy learning module which consists of a convolution layer of size $2 \times 2 \times 64$, and a ConvLSTM [64] unit to incorporate the history. The policy is aimed to learn Q-values corresponding to the k discrete actions.

B. Direct Policy Transfer to the Real World

Our Semantic Net incorporates robust visual features trained on rich semantic object categories and is capable of producing correlation map between the visual goal and the robot observation. Our simple mechanism for computing the visual correlation between the goal and the observation is the crux of our network for modeling domain invariant stimuli $\phi_{\hat{\theta}_s}$. This stimuli is directly fed to our policy network so that the agent can learn generalizable policies that can be directly transferred to the real world.

At the test time, we are able to use the same network for querying our visual servoing policy with semantic object categories. To do that, we mask the last layer of the Semantic Net with the category id l to produce $S_l(I)$ and use it in lieu of the correlation map $\widehat{\phi_{\hat{\theta}_s}}(I, l) = S_l(I)$ which will be fed into our policy module. Given the fact that our Semantic Net is trained with real images and is robust to noisy samples [25], our approach for computing the semantic correlation map as an input to the policy network provides domain invariant representations which we will empirically show to work well in real world scenarios. Our Collision Net is also domain invariant as it is pre-trained via domain randomization technique [50].

C. Object Reaching via Deep RL

We consider a goal conditioned agent interacting with an environment in discrete timesteps. Starting from a random policy the agent is trained to choose actions towards getting closer to a goal. The goal is defined by cropping out the image patch around the goal object as seen at the initial state and is denoted by I^g. At each timestep t, the agent receives an observation o_t, takes an action a_t from a set of k discrete actions $\{a^1, a^2, ..., a^k\}$ and obtains a scalar reward R_t. Each action corresponds to a rotation angle using which a continuous motion vector is computed to move the agent forward in the 3D environment. The motion vector has constant velocity for all the actions. By following its policy π_θ, the agent produces a sequence of state-action pairs $\tau = \{s_t, a_t\}_{t=0}^{T-1}$ after T steps. The goal of the agent is to maximize the expected sum of discounted future rewards with a discounting factor $\gamma \in [0, 1]$:

$$Q(s_t, a) = R(s_t, a) + E_{\tau \sim \pi_\theta}[\sum_{t'=t+1}^{T} \gamma^{t'-t} R(s_{t'}, a_{t'})] \qquad (2)$$

Dense Multi-Step Monte Carlo Rollouts We perform multi-step Monte Carlo policy evaluation [57] for all possible K

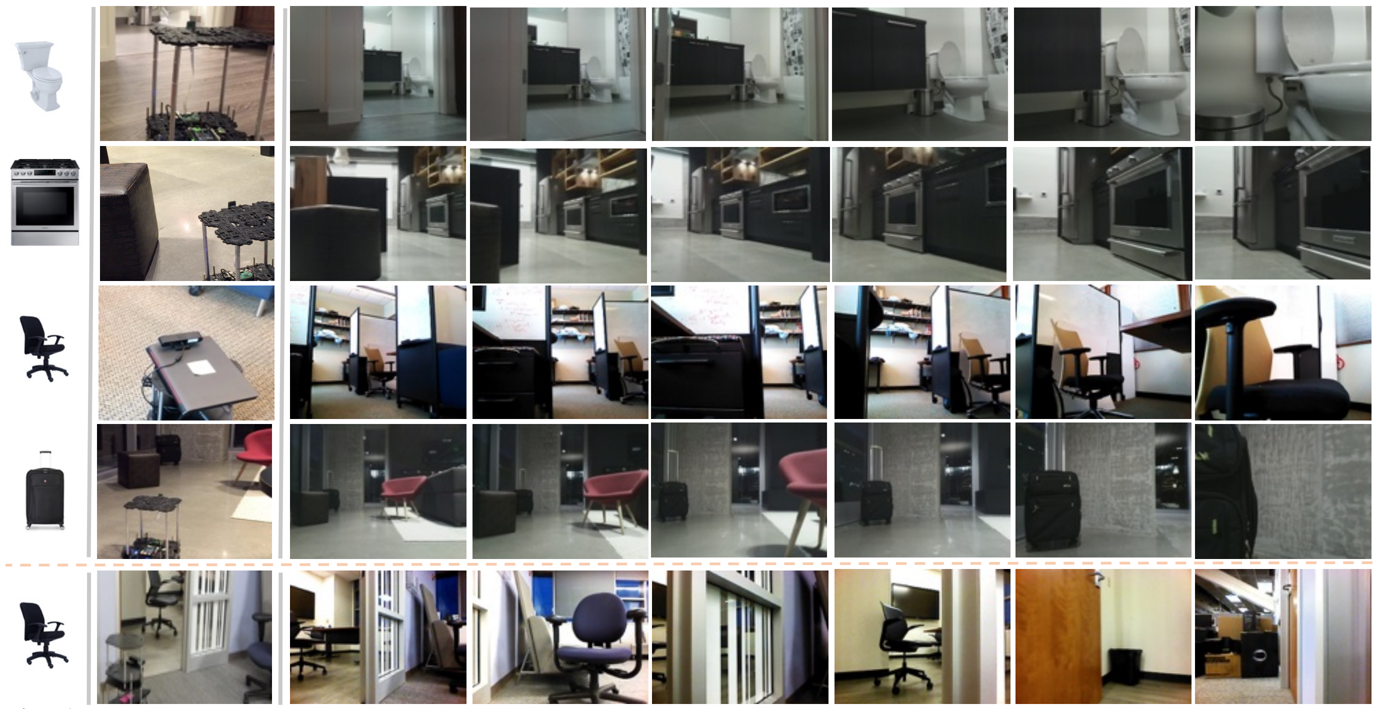
Fig. 5: Real world experiment for reaching semantic goals of toilet, teddy bear, chair and suitcase. The sequences show the robot view captured by head mounted monocular camera. DIViS can generalize to reach semantic goal objects in the real world. Last row shows the failure case where two instances of the queried object are present in the scene.

actions at each state visited during an episode to generate dense rollouts. This enables us to train a deep network to make long-horizon dense predictions. Starting the agent from an initial state we generate dense rollouts with maximum length of T. For each state along the trajectory τ, the dense rollouts are generated by performing $K-1$ additional rollouts corresponding to the actions $\{a^i\}_{i=1,a^i \neq a_t}^{K}$ which are not selected according to the agent's current policy $a_t \sim \pi_\theta$.

Figure 1 demonstrates our dense Monte Carlo rollouts along a trajectory. The agent moves forward based on π_θ. However, policy evaluations are computed for all possible actions that can be taken in each state. The return of each action a is evaluated according to Equation 2. For each state along a trajectory, we compute $\mathbb{Q}_{s_t} = \{Q(s_t, a^i)\}_{i=1}^{K}$ that densely encapsulates Q values quantifying the expected sum of future rewards for each of the possible actions a^i. This policy evaluation provides a dataset of trajectories of the form:

$$(s_0, \mathbb{Q}_{s_0}, a_0, ..., s_{T-1}, \mathbb{Q}_{s_{T-1}}, a_{T-1}) \tag{3}$$

If at any point during the episode or at any of the Monte Carlo branches, the agent collides with any of the objects in the scene the corresponding rollout branch will be terminated.

Batch RL: During training, we use batch RL [32], where we generate dense rollouts with Monte Carlo return estimates as explained above. Starting with a random policy, during each batch the agent explores the space by following its current policy to produce rollouts that will be used to update the policy. For each of the training tasks, we collect dense Monte Carlo rollouts multiple times each with a randomized rendering setup. During training, we collect samples from all our environments and learn a single policy over all different reaching tasks simultaneously. This enforces the agent to learn the common shared aspect between various tasks (i.e. to reach different goals) and acquire generalization capability to unseen test scenarios rather than memorizing a single task seen at the training time.

Reactive Policy: The *reactive* agent starts from a random policy and does not save history from its past observations. The state at each timestep is described by the observation $s_t = o_t$. The visual observation o_t at timestep t is represented by $[\phi_{\hat{\theta}_c,t}, \phi_{\hat{\theta}_s,t}]$ and $(o_t, \mathbb{Q}_{o_t})$ pairs are used to update the policy.

Recurrent Policy: Starting from a random policy, the agent learns a recurrent policy using a sequence of observations. The recurrent policy uses the entire history of the observation, action pairs to describe the state $s_t = (o_1, a_1, ..., a_{t-1}, o_t)$. Each observation along the trajectory is represented by $\Phi_{\hat{\theta}_t}$ according to Equation 1. Given the sequential nature of the problem, we use dense trajectories described in Equation 3 and we model the policy π using a ConvLSTM unit. Intuitively, the history of past observations and actions will be captured in the hidden state of ConvLSTM.

Reward Function: For collision-free goal reaching, the agent should traverse a trajectory that decreases its distance to the goal object while avoiding obstacles. This implies two objectives to be reflected in the reward function: (1) We define the collision reward function as $R_c = \min(1, \frac{d_o - r}{\tau_d - r})$ to penalize the agent for colliding with various objects in the environment. Here, r is the vehicle radius, d_o denotes distance to the closest obstacle, and τ_d is a distance threshold. (2) The agent is rewarded whenever it makes progress towards the goal. Considering d_t as the distance of the agent to the goal and d_{init}

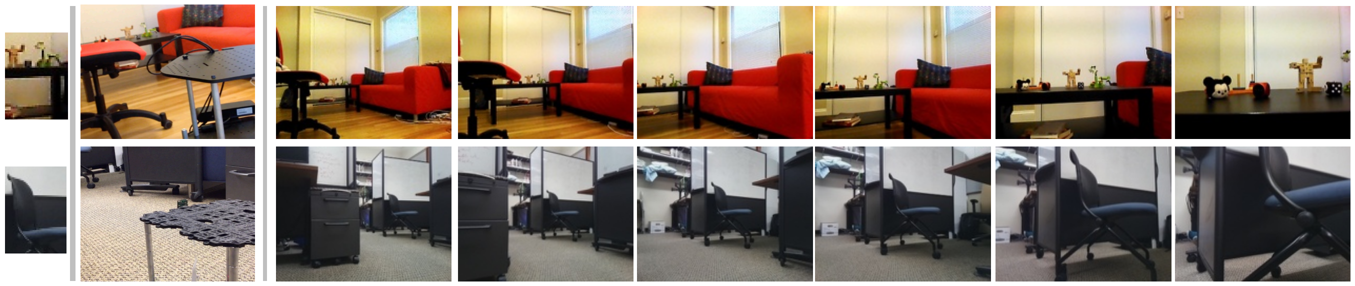

Fig. 6: Real world experiments for reaching a goal in the robots view identified with a goal image. The first and second columns show the goal image and the third person view of the robot respectively. The image sequence show the input RGB images. Our goal reaching policy can generalize to diverse goals and can successfully avoid collisions in real world situations.

as the initial distance of the agent to the goal, our progress reward function is defined as $R_g = \max(0, 1 - \frac{\min(d_t, d_{init})}{d_{init}})$. The total reward is $R = R_c + R_g$.

V. Experimental Results

We evaluate the performance of DIViS for transferring semantic vision-based policy to the real world by conducting real-world experiments with two different real mobile robots. We also, study various design decisions via detailed quantitative simulated experiments and compare against baselines, alternative approaches, and different network architectures.

A. Real-World Evaluations

We use two different mobile robot platforms, TurtleBot2 and Waffle Pi (TurtleBot3), equipped with different monocular cameras (Astra and Raspberry Pi camera) to capture RGB sensory data used as input to our network shown in Figure 1. We compare various settings of training DIViS entirely trained in our domain randomized simulator and without any further fine-tuning or adaptation. Our goal is to answer the following key questions: (1) How well DIViS generalizes to real world settings while no real robot navigation data is used at training time? (2) How well does our approach transfer real world object-level semantics into the policy that is entirely trained in simulation? (3) How effective is our proposed recurrent policy compared to a reactive policy that is trained with similar pre-trained visual features? What is our performance compared to a conventional approach that visually matches the goal with current camera view? We study answer to these questions in the context of quantitative and qualitative real-world experiments.

1) Quantitative Real World Experiments: Our quantitative real world evaluation consists of two different setups for collision-free goal-reaching (a) Goal Image: reaching a visual goal as specified by a user selecting an image patch from the initial robot view. (b) Semantic object: Reaching a semantic object category such as *chair*, *teddy bear*, etc. that is inside the initial robot view.

Generalizing to real world: Table I, shows that our policy is robust to visually diverse inputs. Our Semantic Net, Collision Net and policy can directly transfer to real world for reaching image goals while avoiding obstacles obtaining an average success rate of 82.35% in goal image and 79.24% in semantic object scenarios using two different real robot platforms.

TABLE I: DIViS success rate using two real robot platforms.

Robot	Goal Image		Semantic Object	
	success rate	#trials	success rate	#trials
WafflePi	82.35%	17	85.18%	27
TurtleBot2	81.81%	22	73.07%	26
Total	82.35%	39	79.24%	53

Generalizing to semantic objects: Our experiments outlined in Table I and Table II show that DIViS can successfully generalize to semantic object categories with an average success rate of 79.24% (over 53 trials) although it has not been directly trained for this task. Since our Semantic Net (explained in section IV) is capable of localizing various object categories, it can provide the visual semantic correlation map for our policy network resulting in high generalization capability.

Ablation and comparison with baseline: We compare the performance of DIViS to its reactive version explained in Section IV as well as a Visual goal matching policy that mimics a conventional visual servoing baseline as explained in Section V-B1. To compare each method against DIViS, we run same scenarios with same goal and same initial robot pose. Each section in Table II, compares the methods over similar scenarios. A successful policy should be capable of making turns to avoid obstacles while keeping track of the target object that can go out of the monocular view of the robot in sub-trajectories. Table II outlines that DIViS has the highest success rate in all setups. While the reactive policy can avoid obstacles it fails to reach the goal when it looses track of the objects at turns. Visual goal matching collides with obstacles more frequently as it greedily moves towards the object without considering the path clearance. Having saved the memory of past observations via a recurrent policy, DIViS is able to keep track of the goal object when it gets out of the view and makes better decisions to avoid obstacles.

Comparison to [44] for visual goal reaching: Amongst prior navigation works, the most related paper to our work is [44] which servos to visual goals and is tested via a Turtlebot2 on 8 scenarios in a single environment. [44] does not deal with simulation to real transfer and does not support navigating to semantic goals. We tested DIViS on 92 different scenarios conducted on 20 real world environments with substantially different appearance, layouts and lighting conditions including outdoors. In total, we gained a success rate of 82.35% in “Goal image” scenarios averaged over 39 trials.

TABLE II: Comparing DIViS to baselines in reaching diverse goals in real world

	Goal Image		Semantic object		Total
	success rate	#trials	success rate	#trials	success rate
DIViS(ours)	**83.78%**	37	**75.6%**	41	**79.48%**
DIViS-reactive(ours)	54.04%		43.9%		48.71%
DIViS(ours)	**82.35%**	17	**85.18%**	27	**84.1%**
Visual Goal Matching	35.29%		18.51%		25.0%
DIViS(ours)	**86.66%**	15	**80.0%**	15	**83.33%**
DIViS-reactive(ours)	53.0%		53.53%		53.53%
Visual Goal Matching	40.0%		20.0%		30.0%

2) Qualitative Real World Experiments: Figure 4 visualizes the performance of DIViS in a real wold "semantic goal" scenario where the goal is to reach the "teddy bear". This example demonstrates the importance of incorporating both object semantics and free space reasoning in choosing best actions to find a collision-free path in order to reach the goal object. First row in Figure 4 shows the RGB images observed by the robot and the second row shows the object localization score map for the "teddy bear" as obtained by our Semantic Net. Red arrows in the third row of Figure 4 show the action direction chosen by visual goal matching policy which only incorporates semantic object understanding. Green arrows show the action directions chosen by DIViS. While visual goal matching guides the robot to get close to the goal object, it does not have any mechanism to avoid obstacles and thus fails by colliding into other room furniture. On the other hand, DIViS maintains a collision-free path by choosing actions that both involve object semantics and free space reasoning. During traversal, DIViS may decide to take turns in order to avoid obstacles. This can result in loosing the sight of object for several steps. Being capable of maintaining a short memory, DIViS can turn back to the goal object after avoiding the obstacle.

3) Failure Case: A common failure case of our method is when two instances of the queried semantic object category appear in the scene. An example of such failure case is shown in the last row of Figure 5 where the robot is tasked to reach a "chair" but it is confused by two chairs in the scene and cannot decide which chair it should reach to. This can be addressed by incorporating an instance level object detection network such as [22] which we leave for future work.

More qualitative examples of DIViS for *goal image* and *semantic object* such as "toilet", "teddy bear", "chair" and "suitcase" are provided in Figure 6 and Figure 5. As demonstrated, DIViS can generalize to various real-world scenarios including diverse set of image goals, diverse object categories and various indoor and outdoor environments. Please check supplementary videos at https://fsadeghi.github.io/DIViS for more examples of the DIViS performance on two real robot platforms as well as results in our simulation environment.

B. Simulation Evaluation

To generate simulation test scenarios, we sample free spaces in the environments uniformly at random to select the initial location and camera orientation of the agent. Therefore, the distance to the goal object and the initial view points change from one scenario to another. To further diversify the test scenarios and test the generalization capability, in each test scenario, we do simulation randomization (also known as domain randomization) [50] via textures that were unseen during the training time. During the course of each trial, if distance of the agent to any of the scene objects other than the goal object becomes less than the agents radius (i.e. $\sim 16cm$) a collision event is registered and the trial is terminated.

1) Quantitative Simulation Experiments: For the evaluation criteria, we report *success rate* which is the percentage of times that the agent successfully reaches the visually indicated object. If distance of the agent with the goal object is less than $30cm$, it is registered as success. We report the average success rate over a total of 700 different scenarios involving 65 distinct goal objects collected from train (380 scenarios) and novel test (320 scenario) environments. We compare DIViS (full model with recurrent policy and use of optical flow) against several alternative approaches explained below. Quantitative comparisons are summarized in Table III.

Random Policy: At each step, the agent selects one of the k actions uniformly at random.

Visual Goal Matching: This baseline models a greedy policy that follows an oracle rule of following the path with highest visual similarity to the goal and mimics conventional image-based visual servoing techniques to find the best matching visual features with the goal. Note that this policy uses a high-level prior knowledge about the underlying task while this information is absent for our agent that is trained via RL. Instead, our agent should learn a policy from scratch without any priors. Visual Goal Matching selects one of the actions based on the spatial location of the maximum visual matching score of the visual goal and the current observation. To compute visual correlation map, we use the same Semantic Net pre-trained features used in our network for fairness.

Visual Goal Matching with Collision Avoidance:. This baseline combines prior sim2real collision avoidance method of [50] with conventional visual servoing for following the path with highest visual match to the goal and lowest chance of collision. We incorporate our predicted collision maps to extend Visual Goal Matching policy for better collision avoidance. Using our Collision Net and Semantic Net, we compute the spatial free space map and semantic correlation map for each observation. We sum up these two maps and obtain a single spatial score map that highlights the action directions with highest visual correlation and lowest chance

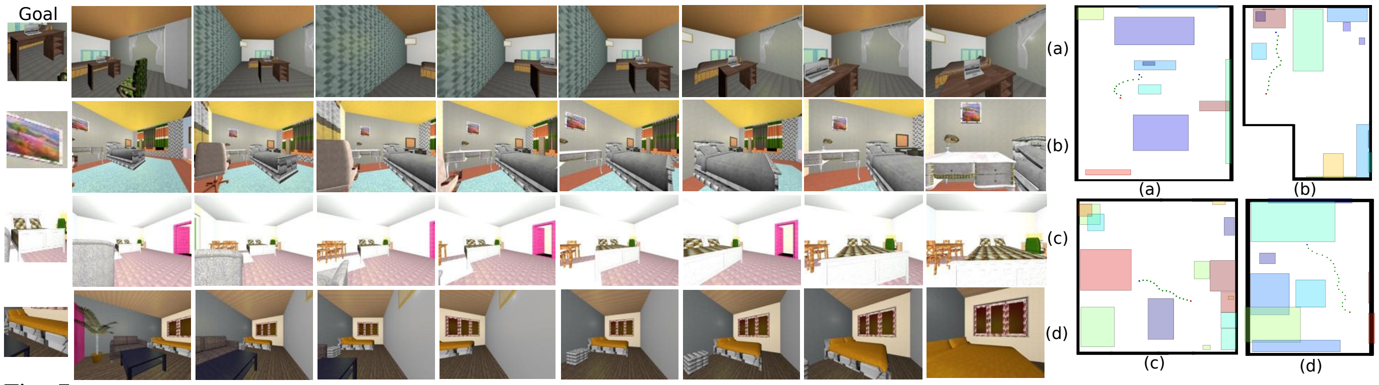

Fig. 7: Qualitative results on several complex test scenarios. In each scenario the trajectory taken by the agent is overlaid on the map (right) and several frames along the path are shown (left). To avoid collisions, the agent needs to take turns that often takes the goal object out of its view and traverse walkable areas for achieving the visually indicated goal in diverse scenarios.

of collision. The agent selects one of the k actions based on the spatial location with highest total score. To be fair, we use the exact same pre-trained features of Collision Net and Semantic Net in our network for this policy.

DIViS - Reactive: The agent selects the best action based on the maximum Q-value produced by the reactive policy explained in Section IV.

DIViS - Recurrent: The agent selects the best action based on the maximum Q-value produced by our recurrent policy without incorporating optical flow ψ explained in Section IV.

DIViS - Recurrent with Optical Flow: Our full model that select the best action based on the maximum Q-value produced by our network that also incorporates optical flow between each two consecutive observations as explained in Section IV and Equation 1.

Table III compares the success rates between different approaches. The highest performance is obtained by our recurrent policy and the best results are obtained when optical flow ψ is also incorporated. While our policy is recurrent, incorporating optical flow further improves the performance. This suggests that optical flow can provide the agent with domain invariant pixel-style motion observations alleviating the difficulty of inference from raw pixels in the presence of domain shift. Interestingly, naive combination of collision prediction probabilities with visual goal matching results in lower performance than only using visual goal matching. This is because the collision avoidance tends to select actions that navigate the agent to spaces with smallest probability of collision. However, in order to reach goals the agent should be able to take narrow paths in confined spaces which might not have the lowest collision probability. Given the results obtained in this experiment, we used our recurrent policy with optical flow during our real-world experiments in Section V-A.

2) Qualitative Simulation Experiments: We qualitatively evaluate the performance and behavior of the best policy i.e. DIViS (with recurrent policy and optical flow) in a number of complex test scenarios. Figure 7 demonstrates several of such examples. In each scenario, the trajectory taken by the agent is overlaid on the top view of map. The initial and final position of the agent are shown by a red and a blue dot, respectively. During these scenarios, the agent needs to take actions which may increase its distance to the goal but will result in avoiding obstacles. However, the agent recovers by taking turns around the obstacles and successfully reaches the goal object.

TABLE III: Success rate in simulation.

Method	Seen Env.	Unseen Env.
DIViS-Recurrent w/flow (ours)	**87.6**	**81.6**
DIViS-Recurrent (ours)	82.1	75.3
DIViS-Reactive (ours)	76.3	69.7
Visual Goal Matching	56.3	54.4
Visual Goal Matching w/ collis.	48.9	47.8
Random policy	23.4	22.2

VI. Discussion

In this paper, we described a novel sim-to-real learning approach for visual servoing which is invariant to the domain shift and is hence called domain invariant. Our proposed domain invariant visual servoing approach, called DIViS, is entirely trained in simulation for reaching visually indicated goals. Despite this fact, DIViS can successfully be deployed on real robot platforms and can flexibly be tasked to reach semantic object categories at the test time and in real environments. Our approach proposes transferring visual semantics from real static image datasets and learning physics in simulation. We evaluated the performance of our approach via detailed quantitative and qualitative evaluations both in the real world and in simulation. Our experimental evaluations demonstrate that DIViS can indeed accomplish reaching various visual goals and semantic objects at drastically different and unstructured real world environments. Our domain invariant visual servoing approach can lead into learning domain invariant policies for various vision-based control problems that involve semantic object categories. Future directions also include investigating how DIViS can be extended to work in dynamic environments such as ones with moving obstacles or goals which is an important challenge to be addressed in real-world robot learning.

Acknowledgement The author would like to thank Emanuel Todorov for great advice and insightful remarks as well as Pieter Abbeel for valuable discussions. The author would also like to thank Maya Cakmak for providing Turtlebot2 at UW for real world experiments. This work was made possible by NVIDIA Graduate Research Fellowship to Fereshteh Sadeghi and Google support to UW.

References

[1] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, 2018.

[2] M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *arXiv preprint arXiv:1808.00177*, 2018.

[3] R. Baillargeon and J. DeVos. Object permanence in young infants: Further evidence. *Child development*, 1991.

[4] R. Basri, E. Rivlin, and I. Shimshoni. Visual homing: Surfing on the epipoles. *IJCV*, 1999.

[5] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, et al. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In *ICRA*, 2018.

[6] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks. In *NIPS*, 2016.

[7] J. G. Bremner. *Infancy*. Blackwell Publishing, 1994.

[8] J. G. Bremner, A. M. Slater, and S. P. Johnson. Perception of object persistence: The origins of object permanence in infancy. *Child Development Perspectives*, 2015.

[9] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 2016.

[10] G. Caron, E. Marchand, and E. M. Mouaddib. Photometric visual servoing for omnidirectional cameras. *Autonomous Robots*, 2013.

[11] F. Chaumette and S. Hutchinson. Visual servo control. i. basic approaches. *IEEE Robotics & Automation Magazine*, 2006.

[12] P. I. Corke. Visual control of robot manipulators–a review. World Scientific, 1993.

[13] M. Cutler, T. J. Walsh, and J. P. How. Real-world reinforcement learning via multifidelity simulators. *IEEE Transactions on Robotics*, 2015.

[14] A. Dosovitskiy and V. Koltun. Learning to act by predicting the future. *arXiv preprint arXiv:1611.01779*, 2016.

[15] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 1992.

[16] D. Fried, R. Hu, V. Cirik, A. Rohrbach, J. Andreas, L.-P. Morency, T. Berg-Kirkpatrick, K. Saenko, D. Klein, and T. Darrell. Speaker-follower models for vision-and-language navigation. *arXiv preprint arXiv:1806.02724*, 2018.

[17] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[19] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *ICCV*, 2011.

[20] A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*, 2013.

[21] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik. Cognitive mapping and planning for visual navigation. *arXiv preprint arXiv:1702.03920*, 2017.

[22] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *CVPR*, 2017.

[23] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, Nguyen, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 2012.

[24] S. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE transactions on robotics and automation*, 1996.

[25] H. Izadinia and P. Garrigues. Viser: Visual self-regularization. *arXiv preprint arXiv:1802.02568*, 2018.

[26] M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*, 2016.

[27] M. Jagersand, O. Fuentes, and R. Nelson. Experimental evaluation of uncalibrated visual servoing for precision manipulation. In *ICRA*. IEEE, 1997.

[28] N. Jakobi, P. Husbands, and I. Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In *European Conference on Artificial Life*. Springer, 1995.

[29] S. James, A. J. Davison, and E. Johns. Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task. *arXiv preprint arXiv:1707.02267*, 2017.

[30] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*. 2012.

[31] T. Lampe and M. Riedmiller. Acquiring visual servoing reaching and grasping skills using neural reinforcement learning. In *IJCNN*, 2013.

[32] S. Lange, T. Gabel, and M. Riedmiller. Batch reinforcement learning. In *Reinforcement learning*. 2012.

[33] A. X. Lee, S. Levine, and P. Abbeel. Learning visual servoing with deep features and fitted q-iteration. *arXiv preprint arXiv:1703.11000*, 2017.

[34] S. Levine et al. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 2018.

[35] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona,

D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[36] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015.

[37] J. Matas, S. James, and A. J. Davison. Sim-to-real reinforcement learning for deformable object manipulation. *arXiv preprint arXiv:1806.07851*, 2018.

[38] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[39] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, et al. Learning to navigate in complex environments. *arXiv preprint arXiv:1611.03673*, 2016.

[40] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*, 2016.

[41] 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*, 2015.

[42] I. Mordatch, K. Lowrey, and E. Todorov. Ensemble-cio: Full-body dynamic motion planning that transfers to physical humanoids. In *IROS*, 2015.

[43] F. Muratore, F. Treede, M. Gienger, and J. Peters. Domain randomization for simulation-based policy optimization with transferability assessment. In *CoRL*, 2018.

[44] D. Pathak, P. Mahmoudieh, G. Luo, P. Agrawal, D. Chen, Y. Shentu, E. Shelhamer, J. Malik, A. A. Efros, and T. Darrell. Zero-shot visual imitation. In *ICLR*, 2018.

[45] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *ICRA*, 2018.

[46] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel. Asymmetric actor critic for image-based robot learning. *arXiv preprint arXiv:1710.06542*, 2017.

[47] R. Pissard-Gibollet and P. Rives. Applying visual servoing techniques to control a mobile hand-eye system. In *ICRA*, 1995.

[48] M. Riedmiller, R. Hafner, T. Lampe, M. Neunert, J. Degrave, T. Van de Wiele, V. Mnih, N. Heess, and J. T. Springenberg. Learning by playing-solving sparse reward tasks from scratch. *arXiv preprint arXiv:1802.10567*, 2018.

[49] A. A. Rusu, M. Vecerik, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell. Sim-to-real robot learning from pixels with progressive nets. *arXiv preprint arXiv:1610.04286*, 2016.

[50] F. Sadeghi and S. Levine. CAD^{2}RL: Real singel-image flight without a singel real image. In *Robotics: Science and Systems*, 2017.

[51] F. Sadeghi, A. Toshev, E. Jang, and S. Levine. Sim2real viewpoint invariant visual servoing by recurrent control. In *CVPR*, 2018.

[52] M. Savva, A. X. Chang, A. Dosovitskiy, T. Funkhouser, and V. Koltun. Minos: Multimodal indoor simulator for navigation in complex environments. *arXiv preprint arXiv:1712.03931*, 2017.

[53] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel. Trust region policy optimization. *CoRR, abs/1502.05477*, 2015.

[54] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[55] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. In *CVPR*, 2017.

[56] G. J. Stein and N. Roy. Genesis-rt: Generating synthetic images for training secondary real-world tasks. In *ICRA*, 2018.

[57] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

[58] M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *JMLR*, 2009.

[59] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT press, 2005.

[60] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *arXiv preprint arXiv:1703.06907v1*, 2017.

[61] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. *arXiv preprint arXiv:1804.06516*, 2018.

[62] W. J. Wilson, C. W. W. Hulls, and G. S. Bell. Relative end-effector control using cartesian position based visual servoing. *IEEE Transactions on Robotics and Automation*, 1996.

[63] Y. Wu, Y. Wu, G. Gkioxari, and Y. Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018.

[64] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NIPS*, 2015.

[65] B. H. Yoshimi and P. K. Allen. Active, uncalibrated visual servoing. In *ICRA*, 1994.

[66] J. Zhang, J. T. Springenberg, J. Boedecker, and W. Burgard. Deep reinforcement learning with successor features for navigation across similar environments. *arXiv preprint arXiv:1612.05533*, 2016.

[67] J. Zhang, L. Tai, Y. Xiong, M. Liu, J. Boedecker, and W. Burgard. Vr goggles for robots: Real-to-sim domain adaptation for visual control. *arXiv preprint arXiv:1802.00265*, 2018.

[68] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.

[69] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. *arXiv preprint arXiv:1609.05143*, 2016.