2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

FlowControl: Optical Flow Based Visual Servoing

Max Argus[1], Lukas Hermann[1], Jon Long[2], Thomas Brox[1]

***Abstract*— One-shot imitation is the vision of robot programming from a single demonstration, rather than by tedious construction of computer code. We present a practical method for realizing one-shot imitation for manipulation tasks, exploiting modern learning-based optical flow to perform real-time visual servoing. Our approach, which we call FlowControl, continuously tracks a demonstration video, using a specified foreground mask to attend to an object of interest. Using RGB-D observations, FlowControl requires no 3D object models, and is easy to set up. FlowControl inherits great robustness to visual appearance from decades of work in optical flow. We exhibit FlowControl on a range of problems, including ones requiring very precise motions, and ones requiring the ability to generalize.**

I. Introduction

The difficulty of robot programming is one of the central hurdles to the widespread application of robots. This task requires domain-specific expertise, making it inaccessible to untrained personnel and resulting in high system costs that lead to low adoption rates. Few-shot imitation from videos is an appealing alternative to overcome this problem, as videos typically capture all task-relevant information. However, the high dimensionality of videos makes it challenging to convert a demonstration video into actionable commands, while at the same time being robust to variations in the environment and the task.

The visual imitation problem can be divided into three separate components:

1) determining the salient objects relevant to the task,
2) establishing correspondences between demonstration and the live application, and
3) controlling the robot in order to reproduce the motion observed in the demonstration.

Each component is a difficult task.

Existing learning-based approaches need large amounts of training data. Other methods that rely on explicit pose estimation require precise 3D models of the objects. Even when given a 3D model, robust 6D pose estimation under appearance variation is an ongoing area of research.

In this paper, we propose a one-shot imitation learning* approach which can robustly replicate a task from a single demonstration video, despite substantial variation of the objects' initial positions, orientations, and appearances. Our approach imitates demonstrations through the use of learned optical flow; point correspondences from optical flow together with a given foreground mask align live observations with demonstration frames. After successfully aligning the live observation with the first demonstration frame, we successively do this for subsequent frames, thereby tracking an entire demonstration trajectory. Thus, this formulation naturally extends to learning multi-step tasks. There is neither a need for CAD models of the objects involved, nor expensive pretraining in elaborate simulation environments.

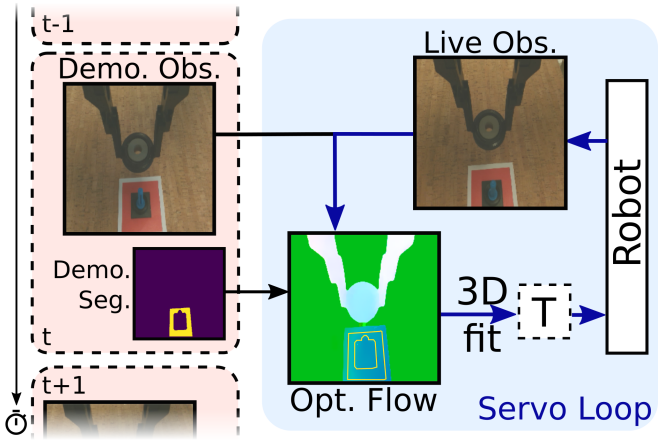


Fig. 1. FlowControl follows demonstrations, shown in red, by using optical flow to find correspondences to a live camera image, shown in blue. This allows us to fit a rigid body transformation T, which aligns the state with the demonstration. The resulting servoing loop is shown with blue arrows. A foreground segmentation provided for the demonstration images (yellow mask) restricts correspondences to the object of interest. By repeating alignment for successive frames, we track trajectories so that tasks can be imitated. Figure 2 shows more details of the transformation computation.

While conceptually straightforward, our approach shows a large degree of robustness towards various factors of variation. We successfully learn a variety of tasks, including picking and insertion of objects. The method is both data-efficient and achieves high success rates.

The main contribution of our work is a practical, data-efficient approach to imitation which exploits and transfers the trained robustness of modern optical flow methods to robot control.

II. Related Work

Imitation learning† tries to control robots in such a way as to replicate what has been demonstrated. This approach reduces or eliminates the need for explicit programming [1].

[1]Computer Science, University of Freiburg, Germany; [2]Symbio Robotics, Emeryville, USA.
Corresponding author: argusm@cs.uni-freiburg.de

*Ours is not a learning method in the sense of machine learning, we do not use data to optimize weights.

†Also know as: Learning from Demonstration, Robot Programming by Demonstration, and Apprenticeship Learning

It is often formulated as a learning problem [2] and numerous strategies to utilize demonstrations exist. These include kinesthetic teaching, the decomposition of movement into motion primitives [3], additional exploration guidance for learning algorithms [4], and the direct learning of input to action mappings.

Some approaches start with low-dimensional inputs, e.g. [5], but most start from high-dimensional sensor input and aim to reduce its dimensionality. This is often done by combining embeddings and imitation. Time Contrastive Networks [6] use multiple perspectives to learn a perspective-invariant embedding, and embeddings of images are used to guide Atari play in [4]. Another approach to the problem is one-shot imitation learning, in which policy networks are fine-tuned on conditioning demonstrations [7]. In contrast, our approach need not learn an encoding to work directly with the high-dimensional sensor input.

Visual Servoing (VS) is the concept of creating a feedback loop in which sensor data is used to compute control commands that change sensor measurements [8], [9]. This feedback actively updates the control based on current observations, allowing for more adaptive control that is robust to variation in the environment [10]. Many tasks in robotics, such as navigation, manipulation and learning from demonstration can be addressed using visual servoing [11], [8], [12]. Visual servoing allows the specification of goal configurations as target feature states to which a control law can servo.

Our approach is a form of visual servoing. Visual servoing can be realized in a number of different ways. It generally consists of three components: (1) image feature extraction, (2) the control law to decide where to move with respect to these features, and (3) joint control to execute this decision. While many works focus on formulating robust control laws and image feature extraction in the context of navigation, we consider the imitation of manipulation tasks. Unlike most other visual servoing techniques ours benefits from the use of robust correspondences to generalize over scene geometry, lighting conditions, and object appearance.

Visual servoing is used in diverse applied robotics fields: aircraft manufacturing [13], robotic surgery [14], marine ROVs [15], and aerial manipulation [16]. Recent methodological extensions range from the combination with pose estimation [17] to servoing to bounding boxes detected by an R-CNN [18].

A combination of visual servoing and optical flow is often used to track poses when servoing with respect to explicit pose estimates, which can be expensive to compute ab initio [19], [20], [21]. Optical flow was used for visual servoing by [22], where flow replaces template matching for visual servoing in an industrial positioning system. In [23], [24], a character navigation policy is learned based on the intermediate representation of optical flow.

A number of recent works combine visual servoing and learning. Some policy learning architectures bear a similarity to visual servoing, e.g. through the use of soft-argmax activations [25] or the use of optical flow as an auxiliary task [26]. A number of applied works have also been published that use a combination of visual servoing and learning-based approaches: [27] trains a model acting as control law based on limited examples for electrical engine construction, [28] uses images as templates, and [29] trains a network to predict relative poses between images. [30] implements target following by learning features and dynamics using reinforcement learning. [31] presented learning-based visual servoing for peg insertion. [32] combines optical flow estimation with policy learning.

The performance of optical flow computation has developed very rapidly in recent years [33], [34]. While initial interest in the optical flow problem was grounded in the context of active motion [35], it is also treated as an independent problem. Good performance of these methods has renewed interest in applications of optical flow [36]. We benefit from the ability of recent learning-based optical flow to generalize, as it has been trained to be robust to common variations of appearance changes.

We use optical flow to solve the dense correspondence problem. Optical flow is commonly defined as the per pixel apparent motion between two consecutive frames, which implies a data distribution. Strictly speaking, we apply an optical flow algorithm outside of its canonical scope. This is not a trivial change since it adversely affects the data distribution; incidence of large displacements and out-of-frame occlusions increases. Learned dense correspondences have also been generated in [37], [38]. Similar to our method, [39], [40] learn to transfer key point affordances used for grasping.

III. FlowControl

The main idea of FlowControl is to align a live video frame with a demonstration consisting of a sequence of target frames; this is illustrated in Figure 2. A learned optical flow algorithm is used to compute correspondences between a live frame and a target frame. Together with the recorded depth this yields a 3D flow of the underlying point cloud, which describes how points in the scene must move to reach the target state. A foreground mask restricts the points to a subset relevant for the task. These foreground points are used for computing the 3D transformation that brings the current frame closer to the target frame, and the robot is moved according to that transformation.

After successfully aligning with the first demonstration frame, the procedure can be successively repeated for subsequent frames. This tracks an entire demonstration trajectory, which can include the manipulation of multiple different objects.

FlowControl benefits from three design choices: (1) the end-of-arm camera setup makes it easy to convert image transformations into the end-effector frame; (2) providing a

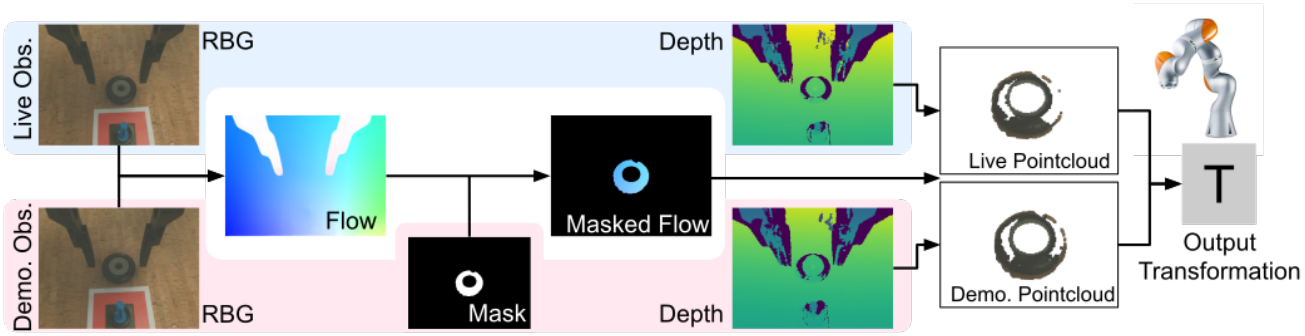


Fig. 2. **FrameAlign** procedure to compute transformations from a live observation to a demonstration image. Optical flow is computed between the two images and masked with the given foreground mask. The points undergo an inverse projection to 3D using the observed depth images. The resulting 3D point correspondences yield the rigid transformation necessary to better align the live state with the demonstration. Due to the large number of correspondences, the method is robust to noisy depth measurement and incomplete foreground masks.

first-person demonstration avoids the problem of a changing perspective; (3) the learned optical flow is robust to many appearance variations.

The foreground mask defines which object in the scene to align in order to progress to the next target state. While such a foreground mask could be inferred automatically from the demonstration video, we simplify the problem by manually providing the foreground mask, trading algorithmic complexity for an intuitive and practical extra step. The foreground mask need not be precise; an under-segmentation of the object suffices.

Optical Flow: For computing correspondences, we use the optical flow from FlowNet2 [34]. FlowNet2 is trained on FlyingThings3D [41], a synthetic dataset procedurally assembled from a large set of different objects. Data augmentation employed in FlowNet training ensures that the resulting optical flow is robust to changes in lighting and partial occlusions. It also performs well on textureless objects, having learned from a large set of different shapes. While we benefit from the robustness and generalization this offers, correspondence is a modular component in our framework, and we could employ other methods such as [42], which does not require any learning. Since the foreground mask is provided for the target image from the demonstration, we compute the optical flow from the target image to the live image, and use this to transfer the segmentation mask directly.

Frame Alignment: We use pixel correspondences from the flow algorithm together with our aligned RGB-D data to match 3D points between the recorded demonstration observation and the live observations. This gives us correspondences between individual points of the pointclouds. Subsequently, we use SVD to compute a least-squares rigid transformation between these point clouds [43]. This is an estimate of the relative transformation between the demonstration scene and the scene as observed live. Servoing in this direction will align the camera image with the demonstration image. We use a position-based controller to convert this transformation into a control signal. Pseudocode for this algorithm is given in Algorithm 1.

Data: demo observation $O^D = (O^D_{\text{RGB}}, O^D_{\text{Depth}})$, live observation $O^L = (O^L_{\text{RGB}}, O^L_{\text{Depth}})$, and demo segmentation S^D.
Result: transformation $T_{R,t}$
`// Compute flow from RGB`
$F = \text{Flow}(O^D_{\text{RGB}}, O^L_{\text{RGB}})$;
`// Compute live segmentation`
$S^L = \text{warp}(S^D, F)$;
`// Apply mask to depth image`
$O'^D_{\text{Depth}}, O'^L_{\text{Depth}} = O^D_{\text{Depth}}[S^D], O^L_{\text{Depth}}[S^L]$;
`// Unproject to 3D pointcloud`
$p^D, p^L = \text{Unproj}(O'^D_{\text{Depth}}, O'^L_{\text{Depth}})$;
`// Fit transformation T`
$T_{R,t} = \underset{R \in SO(3), t \in R^3}{\text{argmin}} \; \Sigma(R \cdot p^D + t) - p^L$;

Algorithm 1: FrameAlign: Procedure to find the transformation T that aligns the current frame with a demonstration frame for the foreground region defined by S^D. Figure 2 depicts this computation.

Sequence Tracking: In order to imitate a complete task, such as grasping an object, we need to successively align with respect to a sequence of demonstration images. To this end, when the live image is sufficiently close to the target image, as defined by a threshold, we step over to the next target image from the demonstration. Our implementation only steps in one direction, so recovering from incorrect actions is not possible.

As converging to each correct alignment takes some time, the tracking process can be accelerated by sampling only every n^{th} demonstration frame. Moving the attention from one object to another requires a new target frame with the foreground mask on the new object.

We need not generalize over some control quantities such as gripper state; for these we just copy the demonstration

actions from the corresponding trajectory step. Interestingly, as long as our method correctly aligns all other dimensions an orthogonal dimension can be copied from the demonstrations. This allows aligning the in-plane components of the relative orientation and copying the height of the gripper.

To account for the time it takes the gripper to close, we delay progressing to the next frame accordingly.

Pseudocode for the sequence tracking is given in Algorithm 2.

Data: demo. observations and segmentations O^D, S^D live observations O^L, demo length N, and distance threshold δ.
Result: imitated interaction
$i = 0$; `// init. demo. frame index`
$T = \infty$; `// init. rel. transformation`
while $i < N$ **do**
  while $|T| > \delta$ **do**
    $T = \text{FrameAlign}(O_i^D, S_i^D, O^L)$; `// get T`
    $a = \text{Action}(O^D, T)$; `// get action`
    $O^L = \text{RobotAct}(a)$
  end
  $i = i + 1$; `// increase demo. frame`
  $T = \infty$;
end

Algorithm 2: Sequence Tracking: Continuously align, and when close enough continue to next demonstration frame.

Task Modularity: Our method can combine individual subtasks into a multi-step task. This is done by switching the object segmented as foreground object during the demonstration. An example is shown in Figure 3, where the first subtask is to grasp the wheel (the foreground mask is on the wheel) before the focus switches to the screw to connect the wheel with the screw.

IV. Experiments

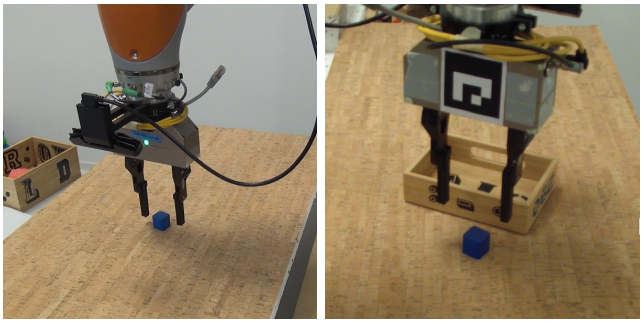
We tested the end-to-end performance of our system on four manipulation tasks and in a localization experiment that is inspired by a real world industrial use case. Since camera pose estimation plays a key role in our setup, we additionally tested this component in isolation and quantified its performance relative to alternative pose estimation methods. Finally we tested the robustness of our system to variations in scene geometry and appearance.

Setup: The experimental setup consists of a KUKA iiwa arm with a WSG-50 two finger parallel gripper and an Intel SR300 projected light RGB-D camera mounted on the flange for an eye-in-hand view. The camera is attached to a 3D-printed mount and faces towards the point between the fingertips. The setup, which is shown in Fig. 3, allows us to record depth images, which we use in our geometric fitting procedure. The movement of the end effector is restricted such that the gripper always faces downwards; it is parameterized as a 5 DoF continuous action $a = [\Delta x, \Delta y, \Delta z, \Delta\theta, a_{\text{gripper}}]$ in the end effector frame. $\Delta x, \Delta y, \Delta z$ specify a Cartesian offset for the desired end effector position, $\Delta\theta$ defines the roll rotation of the end effector, and a_{gripper} is the gripper action that is mapped to the binary command to open or close the fingers.

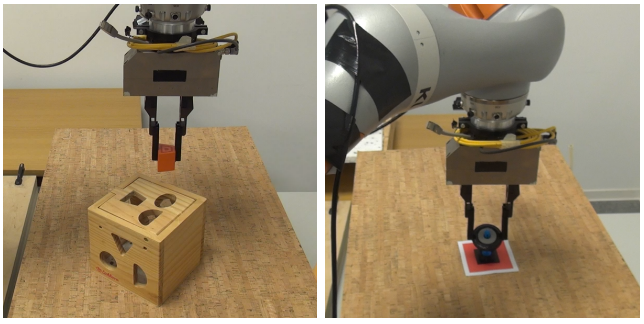
Our state observation consists of 640×480 pixel RGB-D camera images and proprioceptive state vector consisting of the gripper height above the table, the angle that specifies the rotation of the gripper, and the width of the gripper fingers. The optical flow is computed at the same resolution.

A. Manipulation Experiments

Our approach is demonstrated by experiments on four example tasks: grasping a wooden block, inserting a block into a shape sorter, and grasping and inserting a wheel onto a screw. Solving these tasks requires precise movements; for example, an error of 4 mm is enough to make the insertion tasks fail. During the experiments we tested the robustness of our method by varying the object positions. To test the reactiveness of the approach we also moved the objects during task execution and varied the lighting conditions. Examples of this are shown in the supplemental videos.



(a) Grasping Task (b) Pick-and-Stow Task



(c) Shape-Sorter Task (d) Wheel Task

Fig. 3. Images of manipulation tasks being completed, shown from an external perspective.

Grasping Blocks: A small, 25 mm wide, wooden block must be grasped and lifted from a table surface. This task is shown in Fig. 3 (a).

Pick-and-Stow: A small wooden block needs to be grasped and lifted from a table surface to be dropped into a box.

Shape-Sorter: A block must be inserted into a shape-sorting cube. This requires precise positioning due to the tight fit of the opening to the block. We start with the block already grasped. This task is shown in Fig. 3 (c).

Wheel Insertion: This task uses parts from a toy construction set. A wheel must be grasped and inserted onto a screw that is held in a vertical position. This task is shown in Fig. 3 (d).

Results: The success rates for our manipulation experiments are shown in Table I, examples are also shown in the supplemental video. During testing, we used the same position distributions for task between methods as the success rate depends on the variation in the environment. Our method achieves high success rates. In addition, we outperform ACGD [44], a recent approach that uses demonstrations to generate a curriculum for reinforcement learning. In contrast to ACGD, FlowControl does not require a simulation environment for task learning, which is difficult and time consuming to set up.

TABLE I
SUCCESS RATES FOR DIFFERENT MANIPULATIONS TASKS.

Task	Method	Success Rate
Pick-Stow	ACGD [44]	17 / 20
Pick-Stow	(Ours)	19 / 20
Shape-Sorter	(Ours)	8 / 10
Wheel Insertion	(Ours)	9 / 10

Despite the generally good performance, we also identified failure cases. A predictable source of problems were occlusions. These occurred, for example, in the pick-and-stow task when the box occluded the cube.

Starting too far away from a target frame of the demonstration also leads to failure. FlowNet2 works within a given range of displacements. If the target is too far away, the optical flow will not find the correspondence anymore. Especially large rotations are a problem for optical flow. Section IV-C quantifies the robustness of FlowControl in greater detail. As the flow algorithm works for limited displacements in image space, starting demonstrations with the robot further away from the objects allows for coping with bigger displacements, as these appear smaller.

Optical flow is also more likely to fail when confusing background flows are present. This is not prevented by the foreground mask as the flow algorithm receives as input the whole image and information from this may confound the flow computation. When running the controller with high velocities, a single wrong optical flow estimate can move the robot out of the convergent zone. Running the controller at smaller velocities allows such errors to be corrected, reducing the chance of failure on the task.

Other practical examples of failure were due to low illumination combined with fixed exposure times and grasping attempts snapping the object out of the visual field.

B. Navigation Experiment

To demonstrate that the proposed method is directly applicable in an industrial setting, we evaluated a practical localization task. This task is based on an automotive assembly scenario in which a nut-runner must fasten nuts to fix an Engine Control Unit (ECU) into position. The nut-runner must be positioned precisely in order to engage the bolts. Our method can visually align the end-effector resulting in greater robustness to variation in workpiece placement. The task is shown in Fig. 4.

We randomized the initial positions of the end-effector in the camera plane and measured how precisely it can return to the given reference position according to the robot's state-estimation. This resulted in a precision of ± 1 mm, also measured using the robot's state estimation system. This test was repeated five times, with starting positions of up to 8 cm from the target position. In these experiments, our reference image was taken with different lighting.

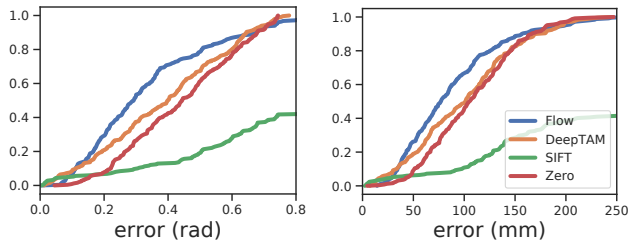
Fig. 4. Navigation task in an industrial setting showing an Engine Control Unit, from the robot's perspective, with respect to which a reference position has to be reached in order to fasten bolts.

C. Fitting Experiment

We evaluated the pose estimation component of our system using a proxy task with pre-recorded images. Instead of moving an object with respect to the camera, we moved the camera with respect to a static object and recorded multiple views. Visual markers were added to the scene to determine the relative camera pose between views; these were calculated using the FreiCalib tool [45]. The pose estimation algorithms must estimate this relative pose. We evaluate several baselines. The simplest is a zero pose change prediction. The SIFT baseline is evaluated similar to [46]. We also compare to DeepTAM [47], a learned algorithm that estimates depth and relative pose given to images.

In this setup, the static background could be used to infer the relative pose. To mitigate this, we again masked our computed features with the demonstration segmentation,

except for DeepTAM, where this was not possible, because it is a monolithic system that produces a pose. As both SIFT matching and optical flow methods have outliers, we substitute zero pose change predictions for rotations larger than $\frac{\pi}{4}$ rad, and translations larger than 250 mm.



Relative Orientation Error Relative Translation Error

Fig. 5. Comparison of the proposed method to baselines showing percent of samples (y-axis) below a given error threshold (x-axis).

This evaluation was done on a selection of 250 views with a relative rotation of less than $\frac{\pi}{4}$ rad. The results are shown in Figure 5. SIFT based relative pose detection performed badly, completely failing for most views. This resulted in worse performance than the zero pose change baseline. DeepTAM performs slightly better than the zero rotation. However, as this approach is designed for smaller pose differences it often underestimates changes. Our flow-based approach performs best for most samples, although it still has outliers in cases where the flow computation failed. An example of this is shown in Figure 6. This usually occurs for a combination of large displacements and rotations.

Fig. 6. Example of erroneous optical flow resulting from a large rotation and limited overlap between the two images. Examples of good flow are shown in Fig. 1 and Fig. 2. White indicates zero flow magnitude.

D. Generalization Experiments

In contrast to classical fixed visual servoing approaches, FlowNet has been trained to be invariant to miscellaneous effects, such as lighting changes and partial occlusion. This helps it find correspondences even when objects in the demonstration do not match exactly. For simplicity, we limited the generalization experiments to the grasping task. We recorded a demonstration with one object and then tested if this demonstration generalizes to objects of different shapes and sizes. Examples of this are shown in Figure 7 and in the supplemental video. FlowControl is able to cope with variation in both color and shape.

V. Discussion and Conclusion

We presented a practical, data-efficient method for visual servoing from optical flow. Our method works with single

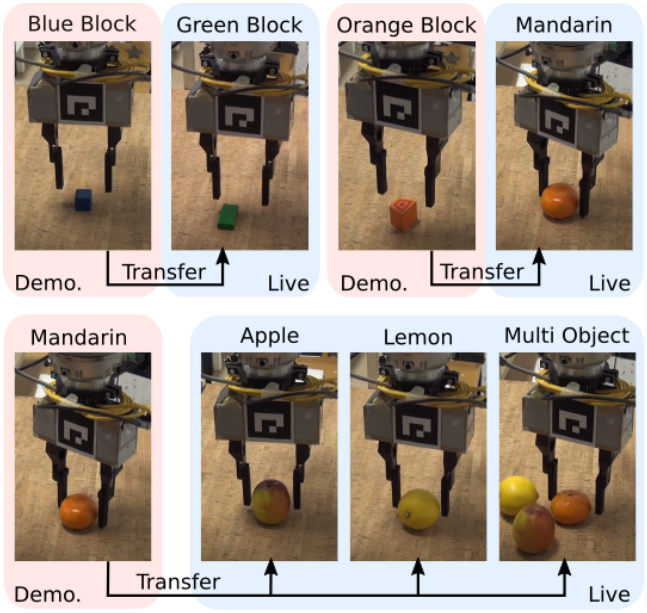


Fig. 7. Generalization experiments showing transfer of grasping with objects that are different from the object used for demonstration.

demonstrations and is able to handle significant variations in the geometric arrangement as well as visual appearance of the task. We demonstrated the effectiveness of our method on a series of robotic manipulation experiments. In addition, we provided a quantitative assessment of the pose estimation part of our algorithm and combined this with a discussion of possible failure cases of our method. Finally, we also provide some experiments indicating that our method is able to generalize over substantial variation in geometry and appearance.

While FlowControl has many advantageous properties, it has natural limitations: it cannot yet do re-grasping and currently relies on manual segmentation to define the task. Current failure cases include optical flow methods failing for large displacements. One could train optical flow specifically for the type of data distribution at hand: one with larger rotations and displacements, or for the specific objects that may appear in the task.

Despite this, FlowControl satisfies an important aim; robotics algorithms should not merely solve one specific task, but instead obviate the need for task-specific engineering. With little manual effort, FlowControl solves a diverse set of tasks.

VI. Acknowledgments

This work was partially funded by the German Research Foundation (DFG) under project number BR 3815/10-1.

References

[1] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Springer Handbook of Robotics*, 2008.

[2] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *Foundations and Trends in Robotics*, vol. 7, pp. 1–179, 2018.

[3] D. Kulic, C. Ott, D. Lee, J. Ishikawa, and Y. Nakamura, "Incremental learning of full body motion primitives and their sequencing through human motion observation," *The International Journal of Robotics Research*, vol. 31, pp. 330 – 345, 2012.

[4] Y. Aytar, T. Pfaff, D. Budden, T. L. Paine, Z. Wang, and N. de Freitas, "Playing hard exploration games by watching youtube," *CoRR*, vol. abs/1805.11592, 2018. [Online]. Available: http://arxiv.org/abs/1805.11592

[5] C. Zimmermann, T. Welschehold, C. Dornhege, W. Burgard, and T. Brox, "3d human pose estimation in rgbd images for robotic task learning," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1986–1992, 2018.

[6] P. Sermanet, C. Lynch, J. Hsu, and S. Levine, "Time-contrastive networks: Self-supervised learning from multi-view observation," *CoRR*, vol. abs/1704.06888, 2017. [Online]. Available: http://arxiv.org/abs/1704.06888

[7] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine, "One-shot imitation from observing humans via domain-adaptive meta-learning," *6th International Conference on Learning Representations, ICLR 2018, Workshop Track Proceedings*, 2018.

[8] D. Kragic and H. I. Christensen, "Survey on visual servoing for manipulation," Computational Vision and Active Perception Laboratory, Tech. Rep., 2002.

[9] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *IEEE Robotics & Automation Magazine*, vol. 13, pp. 82–90, 2006.

[10] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, 2019. [Online]. Available: https://science.sciencemag.org/content/364/6446/eaat8414

[11] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, Oct 1996.

[12] J. Bandera, J. Rodríguez, L. Molina-Tanco, and A. Bandera, "A survey of vision based architectures for robot learning by imitation," *International Journal of Humanoid Robotics*, vol. 9, 03 2012.

[13] A. Kheddar, S. Caron, P. Gergondet, A. Comport, A. Tanguy, C. Ott, B. Henze, G. Mesesan, J. Englsberger, M. A. Roa, P.-B. Wieber, F. Chaumette, F. Spindler, G. Oriolo, L. Lanari, A. Escande, K. Chappellet, F. Kanehiro, and P. Rabate, "Humanoid robots in aircraft manufacturing," *IEEE Robotics and Automation Magazine*, vol. 26, no. 4, pp. 30–45, Dec. 2019. [Online]. Available: https://hal-lirmm.ccsd.cnrs.fr/lirmm-02303117

[14] B. Huang, M. Ye, S.-L. Lee, and G.-Z. Yang, "A vision-guided multi-robot cooperation framework for learning-by-demonstration and task reproduction," *International Conference on Intelligent Robots and Systems (IROS)*, pp. 4797–4804, 2017.

[15] S. Sivčev, M. Rossi, J. Coleman, G. Dooly, E. Omerdić, and D. Toal, "Fully automatic visual servoing control for work-class marine intervention rovs," *Control Engineering Practice*, vol. 74, pp. 153 – 167, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0967066118300327

[16] P. Ramon-Soria, B. C. Arrue, and A. Ollero, "Grasp planning and visual servoing for an outdoors aerial dual manipulator," *Engineering*, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2095809919308653

[17] A. Vakanski, F. Janabi-Sharifi, and I. Mantegh, "An image-based trajectory planning approach for robust robot programming by demonstration," *Robotics and Autonomous Systems*, vol. 98, pp. 241–257, 2017.

[18] B. Joffe, K. Ahlin, A.-P. Hu, and G. McMurray, "Vision-guided robotic leaf picking," EasyChair Preprint no. 250, EasyChair, 2018.

[19] B. J. Nelson, N. Papanikolopoulos, and P. K. Khosla, "Visual servoing for robotic assembly," in *Visual Servoing-Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback*, K. Hashimoto, Ed. River Edge, NJ: World Scientific Publishing Co. Pte. Ltd., 1993, pp. 139–164.

[20] B. J. Nelson, J. D. Morrow, and P. K. Khosla, "Improved force control through visual servoing," *American Control Conference - ACC'95*, vol. 1, pp. 380–386 vol.1, 1995.

[21] M. Vincze, M. Ayromlou, S. Chroust, M. Zillich, W. Ponweiser, and D. Legenstein, "Dynamic aspects of visual servoing and a framework for real-time 3d vision for robotics," in *Sensor Based Intelligent Robots*, G. D. Hager, H. I. Christensen, H. Bunke, and R. Klein, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 101–121.

[22] T. Mitsuda, Y. Miyazaki, N. Maru, K. F. MacDorman, A. Nishikawa, and F. Miyazaki, "Visual servoing based on coarse optical flow," *IFAC Proceedings Volumes*, vol. 32, no. 2, pp. 539 – 544, 1999, 14th IFAC World Congress 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667017560929

[23] A. López, F. Chaumette, E. Marchand, and J. Pettré, "Character navigation in dynamic environments based on optical flow," *Computer Graphics Forum*, vol. 38, no. 2, pp. 181–192, May 2019, eurographics 2019 proceedings. [Online]. Available: https://hal.inria.fr/hal-02052554

[24] A. Lopez, F. Chaumette, E. Marchand, and J. Pettre, "Attracted by light: vision-based steering virtual characters among dark and light obstacles," in *MIG 2019 - ACM SIGGRAPH Conference Motion Interaction and Games*. ACM, 2019, pp. 1–6. [Online]. Available: https://hal.inria.fr/hal-02299397

[25] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learn. Res.*, vol. 17, pp. 39:1–39:40, 2015.

[26] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," *CoRR*, vol. abs/1610.00696, 2016. [Online]. Available: http://arxiv.org/abs/1610.00696

[27] F. Castelli, S. Michieletto, S. Ghidoni, and E. Pagello, "A machine learning-based visual servoing approach for fast robot control in industrial setting," *International Journal of Advanced Robotic Systems*, vol. 14, 2017.

[28] F. Sadeghi, A. Toshev, E. Jang, and S. Levine, "Sim2real viewpoint invariant visual servoing by recurrent control," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2018, pp. 4691–4699.

[29] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, and P. I. Corke, "Training Deep Neural Networks for Visual Servoing," in *ICRA 2018 - IEEE International Conference on Robotics and Automation*. IEEE, 2018, pp. 3307–3314. [Online]. Available: https://hal.inria.fr/hal-01716679

[30] A. X. Lee, S. Levine, and P. Abbeel, "Learning visual servoing with deep features and fitted q-iteration," *CoRR*, vol. abs/1703.11000, 2017. [Online]. Available: http://arxiv.org/abs/1703.11000

[31] J. C. Triyonoputro, W. Wan, and K. Harada, "Quickly inserting pegs into uncertain holes using multi-view images and deep network trained on synthetic data," *CoRR*, vol. abs/1902.09157, 2019. [Online]. Available: http://arxiv.org/abs/1902.09157

[32] A. Amiranashvili, A. Dosovitskiy, V. Koltun, and T. Brox, "Motion perception in reinforcement learning with dynamic objects," in *Conference on Robot Learning*, 2018.

[33] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," *CoRR*, vol. abs/1504.06852, 2015. [Online]. Available: http://arxiv.org/abs/1504.06852

[34] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," *CoRR*, vol. abs/1612.01925, 2016. [Online]. Available: http://arxiv.org/abs/1612.01925

[35] J. J. Gibson, *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates Inc, 1979.

[36] F. Güney, L. Sevilla-Lara, D. Sun, and J. Wulff, "What is optical flow for?: Workshop results and summary," in *Computer Vision – ECCV 2018 Workshops*, L. Leal-Taixé and S. Roth, Eds. Cham: Springer International Publishing, 2019, pp. 731–739.

[37] H. Bristow, J. Valmadre, and S. Lucey, "Dense semantic correspondence where every pixel is a classifier," *CoRR*, vol. abs/1505.04143, 2015. [Online]. Available: http://arxiv.org/abs/1505.04143

[38] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker, "Universal correspondence network," in *Advances in Neural Information Processing Systems 30*, 2016.

[39] P. R. Florence, L. Manuelli, and R. Tedrake, "Dense object nets: Learning dense visual object descriptors by and for robotic manipulation," in *2nd Annual Conference on Robot Learning, CoRL 2018 Proceedings*, 2018, pp. 373–385.

[40] L. Manuelli, W. Gao, P. R. Florence, and R. Tedrake, "kpam: Keypoint affordances for category-level robotic manipulation," *ArXiv*, vol. abs/1903.06684, 2019.

[41] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, arXiv:1512.02134. [Online]. Available: http://lmb.informatik.uni-freiburg.de/Publications/2016/MIFDB16

[42] R. Anderson, D. Gallup, J. Barron, J. Kontkanen, N. Snavely, C. Hernandez, S. Agarwal, and S. Seitz, "Jump: Virtual reality video," *ACM Transactions on Graphics*, vol. 35, pp. 1–13, 11 2016.

[43] A. Horn, "Doubly stochastic matrices and the diagonal of a rotation matrix," *American Journal of Mathematics*, vol. 76, no. 3, pp. 620–630, 1954. [Online]. Available: http://www.jstor.org/stable/2372705

[44] L. Hermann, M. Argus, A. Eitel, A. Amiranashvili, W. Burgard, and T. Brox, "Adaptive curriculum generation from demonstrations for sim-to-real visuomotor control," in *International Conference on Robotics and Automation (ICRA)*, Paris, France, 2020. [Online]. Available: https://ras.papercept.net/proceedings/ICRA20/1627.pdf

[45] C. Zimmermann, A. Schneider, M. Alyahyay, T. Brox, and I. Diester, "Freipose: A deep learning framework for precise animal motion capture in 3d spaces," Department of Computer Science, University of Freiburg, Tech. Rep., 2020. [Online]. Available: "https://lmb.informatik.uni-freiburg.de/projects/freipose/"

[46] I. Skrypnyk and D. G. Lowe, "Scene modelling, recognition and tracking with invariant image features," *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, pp. 110–119, 2004.

[47] H. Zhou, B. Ummenhofer, and T. Brox, "Deeptam: Deep tracking and mapping," *CoRR*, vol. abs/1808.01900, 2018. [Online]. Available: http://arxiv.org/abs/1808.01900