

ForceSight: Multi-Task Text-Guided Mobile Manipulation with Visual-Force Goals

Anonymous Author(s)

Affiliation
Address
email



Figure 1: ForceSight is an RGBD-adapted, text-conditioned vision transformer. Given an RGBD image and a text prompt, ForceSight produces visual-force goals for a mobile manipulator. Action primitives, shown below each image, are appended to the text input by a simple low-level controller. D refers to the estimated depth of the goal location in meters.

1 **Abstract:** Prior work has demonstrated that deep models outputting human-
2 interpretable kinematic goals can enable dexterous manipulation by real robots.
3 Forces are critical to manipulation, yet have typically been relegated to lower-level
4 execution in these systems. We present ForceSight, a system for multi-task, text-
5 guided mobile manipulation with a deep model that outputs visual-force goals that
6 consist of a kinematic goal and a force goal. The kinematic goal defines a target
7 end-effector pose in image space, enabling visual servoing, while the force goal
8 can be achieved using feedback control with grip force and wrist force measure-
9 ments. Given a single RGBD image and a text prompt as input, ForceSight's deep
10 model outputs a human-interpretable visual-force goal. In one evaluation, Force-
11 Sight enabled a mobile manipulator with an eye-in-hand RGBD camera to perform
12 precision grasps, drawer opening, and object handovers with an 81% success rate
13 in unseen environments with object instances that differed significantly from the
14 training data. In a separate experiment, relying exclusively on visual servoing and
15 ignoring force goals dropped the success rate from 90% to 45%, demonstrating
16 that force goals can significantly enhance performance. Videos, code, and trained
17 models are available at <https://force-sight.github.io/>.

18 **Keywords:** Transformers, Imitation Learning, Manipulation, Force Sensing, Lan-
19 guage Grounding

20 1 Introduction

21 Robotic manipulation has significantly benefited from the integration of tactile and force informa-
22 tion. These modalities enable direct perception and control of contact with the environment, which

23 can be advantageous. For example, grasping a small or flat object off a surface can benefit from first
24 making fingertip contact with the surface and then sliding the fingertips across the surface to pick
25 up the object [1]. Success depends on fingertip force that is high enough to maintain contact with
26 the surface and grasp the object, but low enough to slide the fingertips. In general, grip force and
27 applied force provide strong cues that appropriate contact has been achieved for a task.

28 We present ForceSight, a transformer-based, text-conditioned robotic planner that output visual-
29 force goals that enable the human-interpretable execution of tasks in novel environments with unseen
30 object instances. ForceSight uses an RGBD-adapted, text-conditioned vision transformer to encode
31 an RGBD image from a gripper-mounted camera and output a visual-force goal relevant to the
32 current part of the task. A visual-force goal consists of a kinematic goal and a force goal. The
33 kinematic goal specifies a target configuration for the end effector as a 3D position, a yaw angle, and
34 the distance between the fingertips. The force goal specifies a target grip force and a target applied
35 force measured by a wrist-mounted force-torque sensor.

36 We present the following contributions:

- 37 • **Deep Model to Infer Visual-Force Goals:** We present a deep model that infers visual-
38 force goals when given an RGBD image from an eye-in-hand camera and a natural lan-
39 guage prompt, and show that force goals significantly improve performance over visual
40 servoing alone.
- 41 • **System to Perform Real-World Tasks:** We present a system that uses visual-force goals
42 from the deep model to perform a tasks in unseen environments with unseen object in-
43 stances.
- 44 • **Open Source:** We release our code, dataset, and trained models.

45 2 Related Work

46 Imitation learning has been widely adopted in the field of robotics, enabling robots to learn from
47 human demonstrations and execute complex tasks. Recent work has explored imitation learning to
48 create robot policies from combined text and image input [2, 3, 4, 5, 6]. Although language and
49 vision are both rich modalities for specifying and executing a task, practical considerations such
50 as occlusion and depth ambiguity limit the performance of such systems in practice. To address
51 these limitations, our method utilizes force information to ground the visual representations that
52 determine a robotic policy. While some methods [3, 4, 7, 8] learn behaviors in a data-efficient
53 manner, and generalize to object pose via clever data augmentation or explicit object representations,
54 they often come at the cost of being restricted to narrow environments with fixed cameras, and do
55 not test with out-of-distribution examples. At the other end of the spectrum, many imitation learning
56 methods prioritize generality, but require much more data to do so [2, 5]. We believe our method lies
57 between these two regimes, achieving generalization across environments and camera poses while
58 only requiring a modest amount of data collection effort.

59 Several existing works have used kinematic objectives for robotic planning [3, 4, 9, 10, 11]. Despite
60 their effectiveness, these methods come with limitations. These methods are restricted to tabletop
61 environments, and their representation of interactions with the environment is often simplified, e.g.
62 by representing grip as a binary value or by predicting whether a collision will occur rather than
63 directly controlling the nature of contact.

64 Several works have shown evidence of contact and force prediction from visual input, and their
65 effectiveness in facilitating control during object manipulation for both humans and robots [12, 13].
66 Prior works have also performed robotic planning with the use of contact points [14, 15], but are
67 often restricted to grasp generation. Many real-world tasks involve complex interactions with the
68 environment that can significantly benefit from tactile information. This is particularly true for
69 tasks requiring dexterous manipulation, such as grasping small objects, opening doors and drawers,
70 pressing buttons, and twisting knobs [16, 17]. Contact and force are modalities that retain their

71 significance across different environments and with various objects. This makes them more readily
72 generalizable than other modalities such as vision or joint states.

73 We present a method for coarse-to-fine visual-force servoing. The idea of combined vision-force
74 servoing has been explored with both classical and data-driven methods in limited settings such
75 as peg-in-hole and contour following [18, 19]. The conceptualization of coarse-to-fine visual ser-
76 voing has also been proposed in prior work using data-driven methods, where many initial poses
77 lead to a singular “bottleneck pose” [7, 8, 20]. However, these data collection methods were only
78 demonstrated to work for single-step tasks in tabletop environments with seen object instances. Our
79 method instead chains together visual-force goals at hand-specified keyframes, similar to bottleneck
80 poses, to complete more complex tasks. In contrast to methods with similar data collection schemes,
81 ForceSight is capable of composing behaviors to complete multi-step tasks, functions in a variety of
82 environments, and generalizes to objects semantically similar to those has been trained on.

83 Most similar to our work is PerAct [4], which performs language-driven tabletop manipulation using
84 kinematic goals. In contrast to PerAct, ForceSight utilizes force information, makes predictions in
85 a moving camera frame on a mobile manipulator, uses closed-loop visual servoing instead of open-
86 loop motion planning, utilizes pre-training, and works with unseen objects in unseen environments.

87 **3 ForceSight: A Force-Based Robotic Planner**

88 We present a system that represents robotic tasks as sequences of visual-force goals, where a goal
89 includes target fingertip locations in the camera frame, a target grip force, and the net force to be
90 applied by the gripper.

91 **3.1 Representing Visual-Force Goals**

92 We define a visual-force goal G as the combination of a kinematic goal G_K and a force goal G_F .
93 (Figure 2) The kinematic goal, which defines the desired 4-DOF pose (x , y , z , yaw) of the gripper,
94 is parameterized as $G_K = \{C_{xy}, C_z, \psi, W\}$.

95 $C_{xy} \in \mathbb{R}^2$ is the 2D pixel coordinate of the 3D goal position projected onto the input image, and
96 $C_z \in \mathbb{R}$ is the depth estimate of the 3D goal position with respect to the camera frame. Together,
97 C_{xy} and C_z define a 3D point corresponding to the desired gripper location. In order to obtain
98 a prediction for C_{xy} , we introduce a gripper position confidence map $\mathcal{A} \in \mathbb{R}^{H \times W}$, which is a
99 probability distribution in pixel space describing the likelihood that the gripper should move to each
100 pixel location. C_{xy} is obtained by taking $\arg \max_{x,y} \mathcal{A}$, the most likely pixel coordinate associated
101 with the goal position of the gripper. Similarly, to obtain C_z , we also introduce a gripper depth map
102 $D \in \mathbb{R}^{H \times W}$. The depth estimate C_z is obtained by taking the pixel value from the 2D depth map,
103 which is conditioned by the confidence map, at the corresponding pixel $\arg \max_{x,y} \mathcal{A}$. W (gripper
104 width) is the Euclidean distance between the predicted fingertip locations, and ψ is the predicted
105 yaw of the gripper with respect to the camera frame. From these values, we can derive the 4-DOF
106 kinematic goal of the gripper, i.e. the desired gripper position, yaw, and aperture.

107 In addition to kinematic goal, we define a force goal $G_F = \{F_A, F_G\}$. The applied force $F_A \in \mathbb{R}^3$
108 is a vector that represents the desired net force associated with the next step of a task, measured
109 by a force/torque sensor mounted to the wrist of the robot, then transformed into the camera frame.
110 Grip force, F_G represents a scalar value which is determined through a small force estimation neural
111 network as described in Appendix B. Force goals use Newtons as units.

112 Given an RGBD observation and a text prompt, ForceSight predicts visual-force goals G_K and G_F ,
113 one keyframe into the future, with respect to the current camera frame. The output representa-
114 tions for ForceSight and eye-in-hand camera setup make the future predictions more amenable to
115 visual servoing [21], which uses closed-loop control for improved robustness. Moreover, the same
116 keyframe predictions have the potential to be useful to other embodied AI systems, For example,
117 another two-fingered robot can utilize the kinematic and force goals either directly or with adapta-
118 tions. The predicted goals remain consistent in a global frame, irrespective of the camera mounting

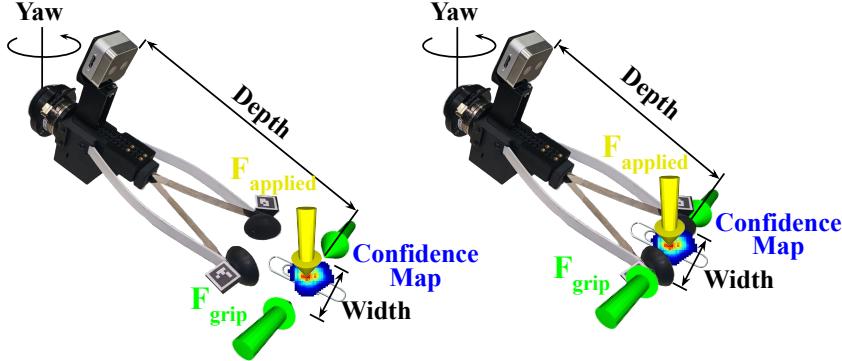


Figure 2: Our representation of visual-force goals includes future fingertip locations (green arrow tips), future grip force (green arrow magnitudes), and future applied force (yellow arrow). The gripper position confidence map represents a probability distribution describing the goal gripper location in pixel space. These goals are represented in the camera frame.

119 location (Figure 5). Consequently, a low-level policy can be adapted to visually servo a robot to
120 reach its predicted goals.

121 3.2 Data Collection

122 We collected a dataset containing over 26,000 high-quality datapoints, each of which consist of an
123 RGBD image, a text prompt, and their corresponding visual-force goal. This dataset, which contains
124 the equivalent of roughly 10,000 task demonstrations, was collected in approximately 30 hours using
125 our data collection method.

126 We accomplish this by collecting data for several perspectives of the same visual-force goal and by
127 associating the same input images with different tasks and their associated goals. For more details,
128 see Appendices B and C.

129 3.3 Architecture

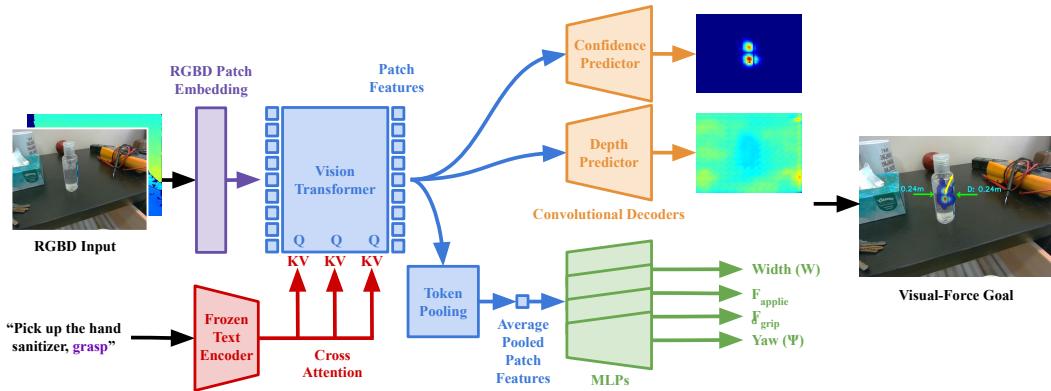


Figure 3: ForceSight is a text-conditioned RGBD vision transformer. An RGBD image is first divided into patches and passed into an RGBD-adapted patch encoder that transforms image patches into visual tokens. These visual tokens are then taken as input into a vision transformer. After every transformer block inside the vision transformer, the visual features are conditioned on text embeddings from a frozen text encoder via cross-attention to produce conditioned patch features. These patch features are passed into two simple convolutional decoders to produce a gripper position confidence map and a gripper depth map. The patch features are additionally average pooled and passed into several MLPs in order to predict the gripper width, applied force, grip force, and yaw.

130 Our proposed architecture, ForceSight, leverages a large-scale Vision Transformer [22] (ViT-large,
131 304M parameters) and a frozen T5 text encoder [23] to output precise visual-force goals (Figure
132 3). Visual-force goals are composed of fingertip locations, grip force, and applied force, enabling
133 robotic manipulation based on tactile objectives.

134 We initialize our vision transformer with weights from pre-training on ImageNet 21k [24]. We in-
135 troduce an enhanced patch embedding layer that accepts RGBD inputs to this pre-trained network.
136 To accommodate depth alongside RGB channels, we add a fourth input channel to the patch embed-
137 ding projection, much like the early fusion technique described in [25]. We initialize the weights
138 of this additional channel with the average of the existing RGB channel weights, enabling smooth
139 integration of depth information without forgetting information learned during pre-training.

140 The text-conditioned component of ForceSight relies on a frozen pre-trained T5 text encoder, which
141 generates text embeddings that offer context for a given task. To effectively utilize this text infor-
142 mation, we incorporate a cross-attention mechanism across all layers of the vision transformer. This
143 approach allows the network to draw rich relationships between text prompts and visual features at
144 multiple levels of abstraction. Inspired by developments in text-conditioned image generation [26],
145 this operation applies cross-attention using learned projections of visual features as query vectors
146 and projections of text features as key and value vectors.

147 The network output includes a 224×224 gripper position confidence map \mathcal{A} and a gripper depth
148 map D . The ViT image encoder generates patch features associated with the input image’s par-
149 ticular regions, which are transformed into a gripper position confidence map and gripper depth
150 map through a convolutional decoder. We apply a weighted cross-entropy loss to the confidence
151 map, translating the task into a pixel-wise classification problem (Equation 4). The depth map pre-
152 diction is supervised via an L_1 loss, masked by the ground truth confidence map, to focus only
153 on the predicted gripper position (Equation 5). Hence, the depth map is conditioned by the confi-
154 dence map, proposing multiple depth estimates C_z in pixel space. Here, we define $C_z = D[C_{xy}]$,
155 where C_{xy} represents the coordinate with the maximum likelihood in the confidence map, given by
156 $C_{xy} = \arg \max_{x,y}(\mathcal{A})$. This representation allows our network to propose multiple hypotheses,
157 exhibiting robustness in complex scenarios where multiple visible objects are semantically relevant
158 to the task.

159 Finally, the patch features are average-pooled and fed into several multi-layer perceptrons (MLPs).
160 These MLPs estimate vital task execution parameters such as gripper width, yaw, applied force, and
161 grip force (Equations 6 - 9).

162 3.4 Action Primitives

163 The use of Large Language Models (LLMs) has demonstrated its efficacy in addressing sequential
164 long-horizon robotic tasks [27, 28, 29, 30, 31]. In language-based tasks, it is often possible to
165 identify shared subgoals across different tasks. By leveraging positive transfer of action subgoals
166 among tasks, it becomes feasible to generalize primitive actions for robot tasks. To facilitate the
167 transition between subgoals, we incorporate action primitives such as *approach*, *grasp*, *ungrasp*,
168 *lift*, and *pull*, which are appended to the model prompt input. The low-level controller switches
169 to the next action primitive once the errors between current states and target goals are adequately
170 minimized, as outlined in the overall system architecture in Figure 12. The inclusion of action
171 primitives in ForceSight has proven effective in facilitating smooth transitions between different
172 subgoals (Table 6). We implement the switching between action primitives using predetermined
173 sequences incremented by a simple state machine, but one could plausibly automate this process
174 with the use of a LLM, as is demonstrated in Appendix J.

175 4 Experimental Results

176 We use a Stretch RE1 [32] from Hello Robot to conduct real-world experiments. To evaluate our
177 model, we conduct experiments in held-out environments, including a mock bedroom and a real

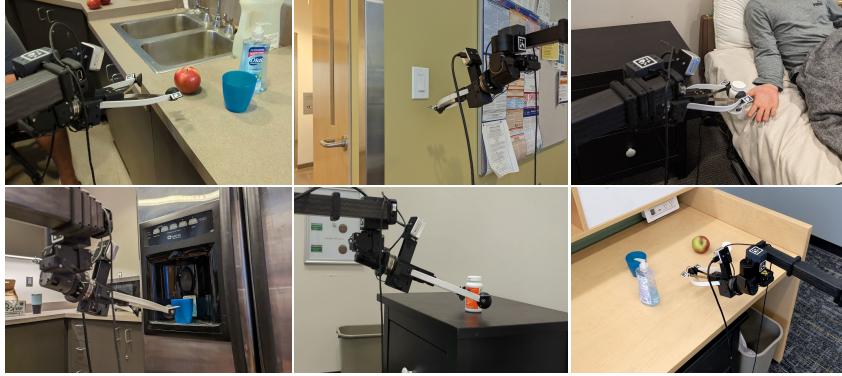


Figure 4: For evaluating ForceSight and conducting ablations in the real world, we select 6 picking tasks, 2 placing tasks, a drawer opening task, and a light switch flipping task. We perform all experiments with the Stretch RE1 robot, encompassing various objects and environments not present in the training set.

178 kitchen (Table 3). We also exclusively interact with unseen object instances, i.e. objects which are
 179 semantically similar to those seen during training but are visually distinct from objects in the training
 180 dataset. We use the low-level controller described in Appendix H to command the robot.

181 4.1 Experimental Setup

182 To evaluate ForceSight, we select a set of 10 household tasks and perform 10 trials on each, totaling
 183 100 trials. We also perform real-world experiments on 5 ablations of our model, each for 20 trials.
 184 For details about task success definitions, see Appendix A. For each trial, we randomize the robot’s
 185 pose within 3 unseen environments such that the target pose is in the camera’s field of view and is
 186 within a distance of 1 meter. We also randomize the pose of the held-out target objects and distractor
 187 objects.

188 We run the model with a lower 8 Hz frequency, providing kinematic and force objectives for visual-
 189 force servoing. Through these experiments, we show that ForceSight achieves a good success rate
 190 given unseen objects and environments.

191 4.2 Metrics

192 For a comprehensive evaluation of ForceSight’s performance, we employ a diverse set of metrics:

193 **Average Contact Distance:** This metric measures the mean distance between the predicted and
 194 actual fingertip locations, provided in meters (m). Smaller values indicate closer matches to the true
 195 locations and thus better prediction accuracy. Given the function inputs, the 3D contact points can be
 196 represented as $C_{L,R} = f(\mathcal{A}, D, W, \psi)$. Importantly, this metric remains consistent across different
 197 output representations, as detailed in Table 6.

198 **Root Mean Square Error (RMSE):** This metric computes the root mean square difference between
 199 the predicted and the actual values of both applied and grip forces ($F_A F_G$), giving insights into the
 200 force estimation’s accuracy.

201 **Task Success Rate:** Expressed as a percentage (%), the Task Success Rate quantifies how often the
 202 system successfully accomplishes the tasks or goals within a set of trials.

203 4.3 Test Set Results

204 We collect a test set in a mock bedroom environment containing several unseen objects (Figure
 205 4). The test set comprises a representative set of 60 keyframe pairs from each task. We evaluate
 206 performance on the test set by calculating the average contact distance, which is the average L_2
 207 error between the predicted locations and the ground truth locations for each fingertip.

	Task Success	Avg. Contact Dist. (m)	RMSE Applied Force (N)	RMSE Grip Force (N)
ForceSight (Ours)	81/100 (81%)	0.036	0.404	1.524
w/o forces	10/20 (50%)	-	-	-
w/o depth	4/20 (20%)	0.063	1.493	1.32
w/o pre-training	4/20 (20%)	0.078	0.583	1.576
w/o augmentation	5/20 (25%)	0.049	1.181	1.759
w/o text conditioning	0/20 (0%)	0.075	0.907	1.260

Table 2: Ablations of ForceSight. Each ablation is tested with 2 trials for each of the 10 real-world tasks for a total of 20 trials per ablation. For more details on success rate see Appendix A.

208 4.4 Real-World Results

Task	Task Success	Subgoal 1	Subgoal 2	Subgoal 3
Pick up the apple	10/10 (100%)	10/10 (100%)	10/10 (100%)	10/10 (100%)
Pick up the medicine bottle	7/10 (70%)	10/10 (100%)	9/10 (90%)	7/9 (78%)
Pick up the keys	6/10 (60%)	9/10 (90%)	9/9 (100%)	6/9 (67%)
Pick up the paperclip	6/10 (60%)	8/10 (80%)	8/8 (100%)	6/8 (75%)
Pick up the hand sanitizer	8/10 (80%)	10/10 (100%)	10/10 (100%)	8/10 (80%)
Pick up the cup	8/10 (80%)	10/10 (100%)	10/10 (100%)	8/10 (80%)
Place object in the trash	10/10 (100%)	10/10 (100%)	10/10 (100%)	-
Place object in the hand	10/10 (100%)	10/10 (100%)	10/10 (100%)	-
Turn off the light switch	7/10 (70%)	9/10 (90%)	7/9 (78%)	-
Open the drawer	9/10 (90%)	10/10 (100%)	9/10 (90%)	9/9 (100%)

Table 1: Real-world task success rates with ForceSight. Tasks are composed of multiple action primitive subgoals. Picking task comprises: *approach, grasp, lift*; placing task: *approach, ungrasp*; turn on light switch task: *approach, push*; and open the drawer task: *approach, grasp, pull*.

- 209 When evaluated on several real-world tasks in novel environments with unseen object instances,
210 ForceSight achieves a task success rate of 81%. According to the results presented in Table 2, we
211 demonstrate the significance of force as a modality for successfully executing tactile tasks, such as
212 picking up a paperclip. Our findings indicate that by using the ForceSight planner, which incorpo-
213 rates force-related objective information, the robot achieves improved performance in these tasks.
214 Conversely, when the low-level controller ignores the force objective information, the robot’s ability
215 to successfully complete the task is compromised. For example, using only kinematic objectives
216 tends to result in items falling due to the lack of grip force consideration. Similarly, neglecting
217 applied forces often results in excessive force or failure to make proper contact with surfaces like
218 tables and drawers, thus worsening grasp success rates.
219 To further investigate the effect of forces on task execution, we performed a more controlled ex-
220 periment comparing the performance of ForceSight with and without force goals. Detailed results
221 are shown in the Appendix 9, 5. From this experiment, we found that ignoring force goals caused
222 the task success rate to drop from 90% (18/20) to 45% (9/20). To summarize our observations, the
223 failure cases predominantly fall into three key categories: 1) the exertion of undue pressure or force
224 on surfaces, 2) challenges related to gripping, and 3) errors associated with positioning.
225 Furthermore, we observed that training the model without depth input leads to a tendency for the
226 robot to prematurely grasp the object. This outcome can be attributed to the lack of depth percep-
227 tion, which affects the robot’s ability to accurately assess the object’s position and make informed
228 grasping decisions. In addition, data augmentation proves to be an effective approach for enhanc-
229 ing performance in diverse environments with unseen objects. In Figure 5, the model also shows
230 generalizability of visual-force goals in unseen instances. We provide more detailed results in the
231 Appendix D.

232 5 Limitations

- 233 While ForceSight exhibits encouraging performance across diverse and challenging tasks, we rec-
234 ognize certain limitations and areas for future improvements in our current methodology.

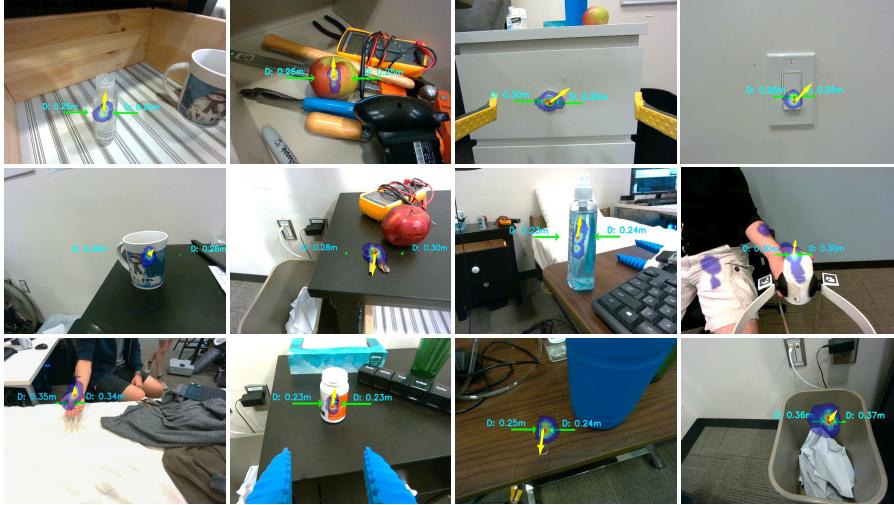


Figure 5: ForceSight is capable of providing accurate estimations of fingertip location, applied forces, and grasping forces associated with a task in a variety of environments. Our network additionally generalizes to unseen object instances, does not depend on a specific camera setup, and works well in situations with partial occlusion.

235 Although we demonstrate efficacy in a set of real-world tasks, our current study still focuses on
 236 a rather limited set of tasks, predominantly consisting of traditional pick-and-place. Given the in-
 237 herent scalability of transformer models and the efficiency of our data collection procedure, we are
 238 optimistic that our methodology could be generalized to a much broader range of complex tasks.

239 One of the recurrent failure modes observed in our model pertains to inaccuracies in depth predic-
 240 tions, particularly those further away from the camera (1m). Despite the adaptive capabilities of
 241 visual servoing help in accounting for some discrepancies as errors diminish at closer distances, we
 242 still plan to explore other depth representations, which may improve the precision of depth predic-
 243 tions and overall task execution.

244 Another limitation of the current ForceSight model is the requirement for targets to be within the
 245 camera’s field of view, which limits performance on some tasks. This constraint may limit task
 246 performance in scenarios such as drawer opening, where predictions are sometimes clipped to the
 247 image’s edge. However, this limitation can be advantageous, as it ensures that the predicted action
 248 goals are grounded within the full observable scene of the camera.

249 While our model is adaptable to various robot manipulators and camera setups, we have only carried
 250 out comprehensive real-world experimental setups using the RE1 robot. To further evaluate the
 251 versatility and robustness of our method, future studies should extend these experiments to various
 252 end-effectors and robotic manipulators.

253 Finally, our keyframe representations do not provide complete information about the gripper’s pose,
 254 as the gripper’s pitch and roll are assumed to be constant values. However, this limitation can be
 255 addressed by adding additional heads to the model’s output to specify these parameters. We believe
 256 that the incorporation of these additional pose parameters will further enhance the performance and
 257 generality of the model.

258 6 Conclusion

259 We presented ForceSight, a text-conditioned robotic planner that generates tactile and kinematic
 260 goals to enable the execution of multiple contact-rich tasks, generalizing to unseen environments
 261 and new object instances. We demonstrated the usefulness of ForceSight with 10 robotic tasks, and
 262 show that the use of tactile objectives improves performance on these tasks.

263 **References**

- 264 [1] V. Babin and C. Gosselin. Picking, grasping, or scooping small objects lying on flat surfaces:
265 A design approach. *The International journal of robotics research*, 37(12):1484–1499, 2018.
- 266 [2] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Haus-
267 man, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv*
268 *preprint arXiv:2212.06817*, 2022.
- 269 [3] M. Shridhar, L. Manuelli, and D. Fox. Cliport: What and where pathways for robotic manipu-
270 lation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.
- 271 [4] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic
272 manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.
- 273 [5] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z:
274 Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learn-
275 ing*, pages 991–1002. PMLR, 2022.
- 276 [6] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess,
277 A. Dubey, C. Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to
278 robotic control. *arXiv preprint arXiv:2307.15818*, 2023.
- 279 [7] E. Valassakis, G. Papagiannis, N. Di Palo, and E. Johns. Demonstrate once, imitate imme-
280 diately (dome): Learning visual servoing for one-shot imitation learning. In *2022 IEEE/RSJ*
281 *International Conference on Intelligent Robots and Systems (IROS)*, pages 8614–8621. IEEE,
282 2022.
- 283 [8] E. Johns. Coarse-to-fine imitation learning: Robot manipulation from a single demonstration.
284 In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 4613–4619.
285 IEEE, 2021.
- 286 [9] T. D. Kulkarni, A. Gupta, C. Ionescu, S. Borgeaud, M. Reynolds, A. Zisserman, and V. Mnih.
287 Unsupervised learning of object keypoints for perception and control. *Advances in neural*
288 *information processing systems*, 32, 2019.
- 289 [10] L. Manuelli, W. Gao, P. Florence, and R. Tedrake. kpam: Keypoint affordances for category-
290 level robotic manipulation. In *Robotics Research: The 19th International Symposium ISRR*,
291 pages 132–157. Springer, 2022.
- 292 [11] S. James, K. Wada, T. Laidlow, and A. J. Davison. Coarse-to-fine q-attention: Efficient learning
293 for visual robotic manipulation via discretisation. In *Proceedings of the IEEE/CVF Conference*
294 *on Computer Vision and Pattern Recognition*, pages 13739–13748, 2022.
- 295 [12] J. R. Flanagan, M. C. Bowman, and R. S. Johansson. Control strategies in object manipulation
296 tasks. *Current opinion in neurobiology*, 16(6):650–659, 2006.
- 297 [13] A. Jain and C. C. Kemp. Improving robot manipulation with data-driven object-centric models
298 of everyday forces. *Autonomous Robots*, 35:143–159, 2013.
- 299 [14] Z. Xue, J. M. Zoellner, and R. Dillmann. Grasp planning: Find the contact points. In *2007*
300 *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 835–840. IEEE,
301 2007.
- 302 [15] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox. Contact-graspsnet: Efficient 6-dof
303 grasp generation in cluttered scenes. In *2021 IEEE International Conference on Robotics and*
304 *Automation (ICRA)*, pages 13438–13444. IEEE, 2021.
- 305 [16] P. Grady, J. A. Collins, S. Brahmbhatt, C. D. Twigg, C. Tang, J. Hays, and C. C. Kemp. Visual
306 pressure estimation and control for soft robotic grippers. In *2022 IEEE/RSJ International*
307 *Conference on Intelligent Robots and Systems (IROS)*, pages 3628–3635. IEEE, 2022.

- 308 [17] J. A. Collins, C. Houff, P. Grady, and C. C. Kemp. Visual contact pressure estimation for
309 grippers in the wild. *arXiv preprint arXiv:2303.07344*, 2023.
- 310 [18] J. Baeten, H. Bruyninckx, and J. De Schutter. Integrated vision/force robotic servoing in the
311 task frame formalism. *The International Journal of Robotics Research*, 22(10-11):941–954,
312 2003.
- 313 [19] K. Almaghout, R. A. Boby, M. Othman, A. Shaarawy, and A. Klimchik. Robotic pick and
314 assembly using deep learning and hybrid vision/force control. In *2021 International Conference
315 “Nonlinearity, Information and Robotics”(NIR)*, pages 1–6. IEEE, 2021.
- 316 [20] B.-S. Lu, T.-I. Chen, H.-Y. Lee, and W. H. Hsu. Cfvs: Coarse-to-fine visual servoing for 6-dof
317 object-agnostic peg-in-hole assembly. *arXiv preprint arXiv:2209.08864*, 2022.
- 318 [21] S. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE transactions
319 on robotics and automation*, 12(5):651–670, 1996.
- 320 [22] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. De-
321 ghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transform-
322 ers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- 323 [23] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu.
324 Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of
325 Machine Learning Research*, 21(1):5485–5551, 2020.
- 326 [24] T. Ridnik, E. Ben-Baruch, A. Noy, and L. Zelnik-Manor. Imagenet-21k pretraining for the
327 masses. *arXiv preprint arXiv:2104.10972*, 2021.
- 328 [25] G. Tzifas and H. Kasaei. Early or late fusion matters: Efficient rgb-d fusion in vision trans-
329 formers for 3d object recognition.
- 330 [26] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image syn-
331 thesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer
332 Vision and Pattern Recognition*, pages 10684–10695, 2022.
- 333 [27] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg. Text2motion: From natural language
334 instructions to feasible plans. *arXiv preprint arXiv:2303.12153*, 2023.
- 335 [28] S. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor. Chatgpt for robotics: Design principles
336 and model abilities. *Microsoft Auton. Syst. Robot. Res*, 2:20, 2023.
- 337 [29] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan,
338 K. Hausman, A. Herzog, et al. Do as i can, not as i say: Grounding language in robotic
339 affordances. *arXiv preprint arXiv:2204.01691*, 2022.
- 340 [30] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson,
341 Q. Vuong, T. Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint
342 arXiv:2303.03378*, 2023.
- 343 [31] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch,
344 Y. Chebotar, et al. Inner monologue: Embodied reasoning through planning with language
345 models. *arXiv preprint arXiv:2207.05608*, 2022.
- 346 [32] C. C. Kemp, A. Edsinger, H. M. Clever, and B. Matulevich. The design of stretch: A com-
347 pact, lightweight mobile manipulator for indoor human environments. In *2022 International
348 Conference on Robotics and Automation (ICRA)*, pages 3150–3157. IEEE, 2022.
- 349 [33] Intel® Realsense™ D405 – intelrealsense.com. <https://www.intelrealsense.com/depth-camera-d405/>.

- 351 [34] ATI Industrial Automation. F/T Sensor: mini45, 2022. URL [https://www.ati-ia.com/
352 products/ft/ft_models.aspx?id=mini45](https://www.ati-ia.com/products/ft/ft_models.aspx?id=mini45).
- 353 [35] M. Mason. The mechanics of manipulation. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 544–548. IEEE, 1985.
- 355 [36] T. Lozano-Perez, M. T. Mason, and R. H. Taylor. Automatic synthesis of fine-motion strategies
356 for robots. *The International Journal of Robotics Research*, 3(1):3–24, 1984.
- 357 [37] A. Majumdar and R. Tedrake. Funnel libraries for real-time robust feedback motion planning.
358 *The International Journal of Robotics Research*, 36(8):947–982, 2017.
- 359 [38] M. Liu, M. Zhu, and W. Zhang. Goal-conditioned reinforcement learning: Problems and
360 solutions. *arXiv preprint arXiv:2201.08299*, 2022.
- 361 [39] J. A. Collins, P. Grady, and C. C. Kemp. Force/torque sensing for soft grippers using an
362 external camera. *arXiv preprint arXiv:2210.00051*, 2022.
- 363 [40] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint
364 arXiv:1412.6980*, 2014.

365 **Appendix**

366 **A Task Definition**

- 367 In our real-world experimentation, we evaluated 10 household tasks, each defined by sequences of
368 action primitives. To determine the success rate for each task, we establish clear criteria for what
369 constitutes a successful completion for each task and subtask.
- 370 We define success for each subtask by whether the kinematic and force errors meet criteria specified
371 in the low-level controller, as is described in Appendix H.
- 372 For tasks that involve picking (e.g. “*pick up the apple*”), the robot should successfully approach,
373 grasp, and lift the target object above the surface for more than 5 seconds.
- 374 For placing tasks (e.g. “*place the object in the hand*”), the robot should successfully approach the
375 desired target object, ungrasp and place the held object to the target object. In this case, into a trash
376 can or a static human hand.
- 377 For the “*turn off the light switch*” task, the light switch should be completely toggled down to be
378 considered a success.
- 379 Additionally, for “*open the drawer*” task, the robot must successfully pull the drawer open, extending
380 to its full range of motion.
- 381 Finally, we label the execution of a task a failure if the trial does not terminate within a 1-minute
382 time window.

383 **B Data Collection Setup**

- 384 In order to collect visual data, we mount an Intel® RealSense™ D405 [33] to the robot’s gripper to
385 capture the RGBD image I . The fingertips are first localized in the image via ArUco tags attached
386 to the gripper, and a transformation is applied to map these to the point at the center of the fingertip
387 surface, which we call the fingertip locations, denoted as $C_{L,R} \in R^{2 \times 3}$ (left and right). Instead of
388 using fiducial markers, this can be readily replaced with other detection methods, such as NN-based
389 object detection. Subsequently, the 3D fingertip contact locations are then mapped from the future
390 camera frame to the current frame by utilizing the robot’s forward kinematics. We use a Stretch
391 mobile manipulator for data collection. Certainly, this can also be substituted with different data
392 collection setups as long as the ground truth future fingertip location can be obtained.
- 393 The force applied to the gripper is measured by a wrist-mounted force/torque sensor [34]. This
394 sensor provides accurate readings of the forces exerted by the gripper, enabling our system to predict
395 future forces and enabling the robot to perform tasks requiring force control.
- 396 The grip force measurement is not natively available on the Stretch robot gripper. To obtain the
397 grip force, we trained a small neural network, parameterized as an MLP, to estimate the grip force
398 F_G given the gripper motor state and fingertip positions. To provide ground truth for the grip force
399 model, we grasp a force/torque sensor [34] at various grip strengths and grasp widths and record the
400 measured magnitude of the force.

401 **C Data Collection details**

- 402 We collect a dataset D , with $D_i = \{I, T, C_{L,R}, F_G, F_R\}$, where $I \in \mathbb{R}^{H \times W \times 4}$ is an RGBD image
403 captured by a gripper-mounted camera, T is a text prompt associated with a task and includes an
404 action primitive, and $C_{L,R} \in \mathbb{R}^{2 \times 3}$ is a set of two 3D fingertip locations in the camera frame
405 associated with the next keyframe. Grip force F_G and Applied force F_A are described in Section
406 3. These combined elements constitute the data points in the dataset, providing a comprehensive
407 representation of a robot’s interaction with its environment. From these datapoints, the kinematic
408 and force goals (G_K and G_F) that serve as ground truth for our network can be derived.

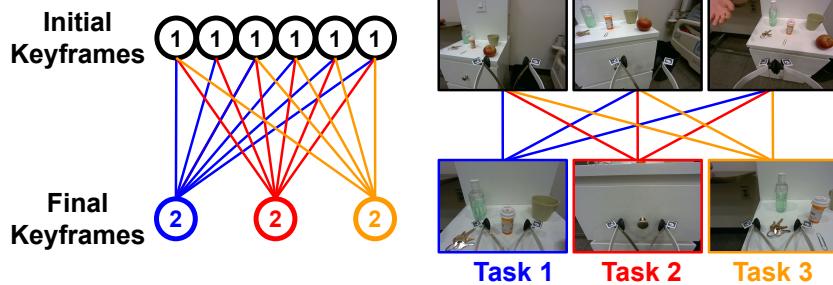


Figure 6: **Left:** During data collection, we partition each task into pairs of hand-specified keyframes. During each data collection session, we pair keyframes associated with several initial gripper poses to a single final keyframe. **Right:** We are also able to pair these initial keyframes with final frames associated with additional tasks.

409 To collect training data for our system, an expert human operator controls the robot using a key-
 410 board in order to capture RGBD images, text prompts, action primitives, and their corresponding
 411 visual-force goals. The kinematic goals and force goals that comprise a visual-force goal are mapped
 412 from the future camera frame to the current frame by utilizing the robot’s forward kinematics. For
 413 efficiency, we had the robot associate multiple inputs with the same goal, collecting views of the
 414 goal from different perspectives. We also sampled from states that are unlikely to occur during a
 415 successful execution in order to promote recovery from errors. For example, tasks containing the
 416 “grasp” action primitive are trained with examples of inputs that are far from the final states associ-
 417 ated with the “approach” action primitive. This enables the “grasp” action primitive to recover from
 418 a poor “approach” or changes due to a dynamic environment. This method allows our algorithm to
 419 be resilient to potentially imprecise predictions, thereby strengthening the robustness of the system.
 420 This is related to the concepts of funnels [35] and pre-image backchaining [36] that have inspired
 421 recent work in robust feedback motion planning [37].

422 To further enhance the efficiency of data collection, items relevant to other tasks were deliberately
 423 included in many initial frames. This strategy allows us to map identical input images to a final
 424 keyframe for each task relevant to the input image, consequently generating substantially more data
 425 points per image in the dataset. Moreover, this has the dual benefit of bolstering the robustness of
 426 text conditioning by exposing the network to negative examples.

427 Our data collection approach, therefore, not only enables a more streamlined and efficient process,
 428 but also exposes our algorithm to a diverse set of environments and object configurations. We believe
 429 that this data collection methodology was key to achieving the necessary robustness to generalize
 430 across a wide range of real-world tasks and environments. Conducted over the course of 30 hours,
 431 our data collection process yielded over 26,000 high-quality keyframe pairs, the equivalent of ap-
 432 proximately 10,000 distinct task demonstrations.

433 Comparing Our Data Collection Method to Sequential Demonstrations

434 We estimate that we collected the equivalent of approximately 10,000 task demonstrations in 30
 435 hours. In pilot experiments, we measured the time it takes to collect keyframes sequentially, coming
 436 up with a lower-bound estimate of 1 minute per demonstration. Assuming this value, collecting
 437 10,000 task demonstrations would have taken approximately $10,000/60 = 167$ hours, as opposed
 438 to the 30 hours we dedicated to collecting data.

439 D Details on Real-world Experiments

440 The real-world experiments for ForceSight are conducted in unseen environments and with held-out
 441 objects (Figure 7 and Table 3). The action primitive sequences are also unique to each task. For
 442 example, the “pick up the apple” task shown in Figure 8 consists of “approach”, “grasp”, and
 443 “lift” action primitives.



Figure 7: **Left:** Objects present in the training set. **Right:** Objects present in the test set and ablation studies.

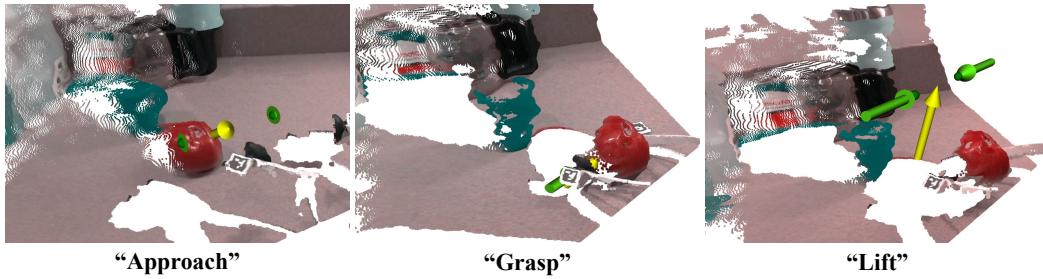


Figure 8: Force objectives prediction in 3D perspective when conducting “pick up the apple” task in a kitchen environment. Yellow arrow represents applied force F_A and green arrows represent grip force F_G acting on the target object.

Task	Environment	Objects	Action Primitives
Pick up the apple	Kitchen	Apple, top drawer	Approach, Grasp, Lift
Pick up the medicine bottle	Bedroom	Medicine bottle, top drawer	Approach, Grasp, Lift
Pick up the keys	Bedroom	Keys, top drawer	Approach, Grasp, Lift
Pick up the paperclip	Bedroom	Paperclip, top drawer	Approach, Grasp, Lift
Pick up the hand sanitizer	Kitchen	Hand sanitizer, counter top	Approach, Grasp, Lift
Pick up the cup	Kitchen	Cup, counter top	Approach, Grasp, Lift
Place object in the trash	Bedroom	Medicine bottle, trash bin	Approach, Ungrasp
Place object in the hand	Bedroom	Medicine bottle, real human hand	Approach, Ungrasp
Turn off the light switch	Atrium	Light switch	Approach, Push
Open the drawer	Bedroom	Bedside drawer	Approach, Grasp, Pull

Table 3: Tasks, environments, objects and success metrics

	Overall Task Success	Subgoal 1	Subgoal 2	Subgoal 3
ForceSight (Ours)	81/100 (81%)	96/100 (96%)	93/96 (97%)	54/65 (83%)
w/o forces	10/20 (50%)	13/20 (65%)	11/13 (85%)	6/7 (86%)
w/o augmentation	5/20 (25%)	11/20 (55%)	7/11 (64%)	5/7 (71%)
w/o depth	4/20 (20%)	8/20 (40%)	6/8 (75%)	3/3 (100%)
w/o pre-training	4/20 (20%)	5/20 (25%)	4/5 (80%)	3/3 (100%)
w/o text conditioning	0/20 (0%)	9/20 (45%)	0/9 (0%)	-

Table 4: Detailed task success rate for ForceSight ablation on real-world experiments.

444 From the analysis presented in Table 4, it is evident that ForceSight outperforms other approaches in
 445 terms of task success rate. Specifically, when the low-level controller disregards the force goals, both
 446 *subgoal 1* and *subgoal 2* exhibit lower success rates. This can be attributed to the robot’s inability to
 447 accurately approach and grip the target object. For example, the gripper will prematurely grasp the
 448 drawer’s handle without applying force to the drawer during the approach stage. Additionally, the
 449 absence of data augmentation adversely affects the model’s ability to perceive new objects, as evi-
 450 denced by its struggles in recognizing unseen objects such as the green apple and the black drawer.
 451 By excluding depth information from the model, the success rate of *subgoal 1* decreases, primarily
 452 because this stage (“approach” action) heavily relies on accurate depth perception. Moreover, pre-
 453 training proves to be a valuable technique for enabling the robot to learn with minimal data. When
 454 our ViT was initialized with random weights rather than pretrained weights, the confidence map
 455 prediction was worsened, resulting in a lower success rate with the same training configurations. In-
 456 terestingly, when text conditioning is removed, the model can successfully detect interesting objects
 457 as goals (9 out of 20 in *subgoal 1*), but fails to determine the appropriate actions to perform with the
 458 target object, seen in *subgoal 2*.

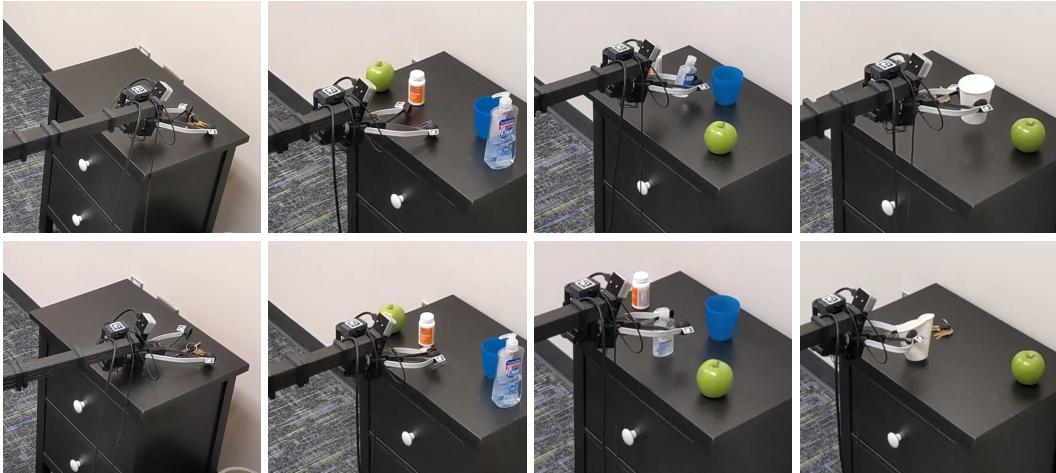


Figure 9: ForceSight with force goals (top) vs. ForceSight without force goals (bottom). Comparison videos are shown on the [website](#).

	Overall Task Success	Subgoal 1	Subgoal 2	Subgoal 3
with forces (ForceSight)	18/20 (90%)	95%	90%	76%
w/o forces	45%	80%	45%	29%

Table 5: Detailed evaluation of task success rate v.s. presence of force as an input modality to the low-level controller.

459 In Table 5, we present a comprehensive ablation study centered on the employment of force In
 460 general, using force as a modality leads to generally improved task performance, especially in tasks

461 that require tactile sensitivity, such as picking up a paperclip. When the controller does not consider
 462 force, the number of errors and misjudgments increases, especially when handling delicate tasks.
 463 The robot often exerted too much force on the object, or the gripper failed to grip. Task such
 464 as "place object in the trash" was not affected since it involves less tactile interactions with the
 465 surroundings.

466 **E Emergent Properties**

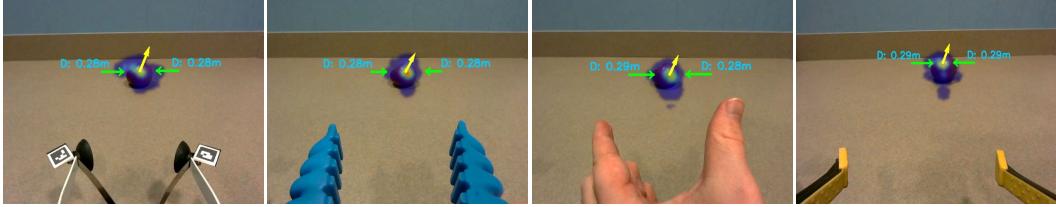


Figure 10: Predictions from ForceSight are agnostic to the agent and camera perspective, as shown in this example for the apple grasping task.



Figure 11: We observe that ForceSight is able to plan more than one keyframe into the future, despite having been trained to predict goals associated with the next keyframe.

467 After training, we observed multiple surprising behaviors exhibited by ForceSight. Notably, Force-
 468 Sight demonstrated a surprising flexibility and adaptability in its treatment of action primitives in
 469 relation to various objects. For instance, ForceSight managed to apply certain action primitives to
 470 objects without any explicit representation of such actions during the training phase. For example,
 471 if given the action primitive "grasp" in combination with a prompt concerning a light switch or the
 472 action primitive "push" in combination with a prompt concerning a medicine bottle, the model re-
 473 sponded with predictions corresponding to the specified actions, despite these specific actions being
 474 absent in the training data. Emergent properties are shown in Figure 10 and 11.

475 We also observe that ForceSight is agnostic to the gripper it is observing; although during training we
 476 only show the network a single gripper, it learns to completely ignore the visual features associated
 477 with the gripper, and the predictions were observed to be invariant to the type of gripper the RGBD
 478 camera was mounted to, and even works without any gripper present in the field of view.

479 **F Determining Output Representation**

480 We experimented with three output representations of the fingertip locations, namely: Regression-
 481 based, Pixel-Space Contact points, and Pixel-Space Centroid output representation.
 482 The regression-based representation directly minimizes the L_1 error of the contact point locations.
 483 We observed this representation to work well in simplified environments with a single object, but
 484 it struggled in situations where there were multiple plausible contact points, especially when two

Representation	Avg. Contact Dist (m)
Contact point regression	0.429
Contact point classification (cls.)	0.107
Centroid cls. + gripper width + yaw	0.057
Centroid cls. + gripper width + yaw + Action Primitive	0.036

Table 6: Representation Ablations.

485 semantically relevant objects were present in the input image. We hypothesize that this is because
486 the L_1 objective encourages the model to find the mean of the target distribution of plausible fingertip
487 locations. In this situation where this distribution is multimodal, the model produces estimates that
488 average out the different modes of the distribution, resulting in a model that provides suboptimal
489 predictions for all modes.

490 To tackle this issue of estimating 3D locations in a multimodal distribution, we instead formulated
491 our optimization as a classification problem. To additionally take advantage of the knowledge
492 present in our pretrained ViT, we decouple the representation into pixel-wise classification and
493 depth-wise regression. This has the additional benefit of highly interpretable predictions, as the
494 confidence map is a distribution that may be overlayed on the input image. We initially treated contact
495 points individually, but found that locations associated with the left and right fingertip didn't neces-
496 sarily correspond to one another, often leading to physically implausible predictions due to the lack
497 of correspondence between fingertip locations.

498 To address this correspondence issue, we reparameterized our representations to instead classify
499 the pixel location associated with the center of the contact points C_{xy} , perform regression on the
500 depth estimates C_z associated with the pixel location, and additionally predict the gripper width and
501 the yaw ψ so that the individual fingertip locations could be derived. The ablation of the output
502 representation is shown in Table 6. A point to note that our current model does not factor in roll and
503 pitch – we have opted to keep roll and pitch constant in our method. That said, roll and pitch can be
504 easily included by adding additional MLP heads.

505 G System Architecture

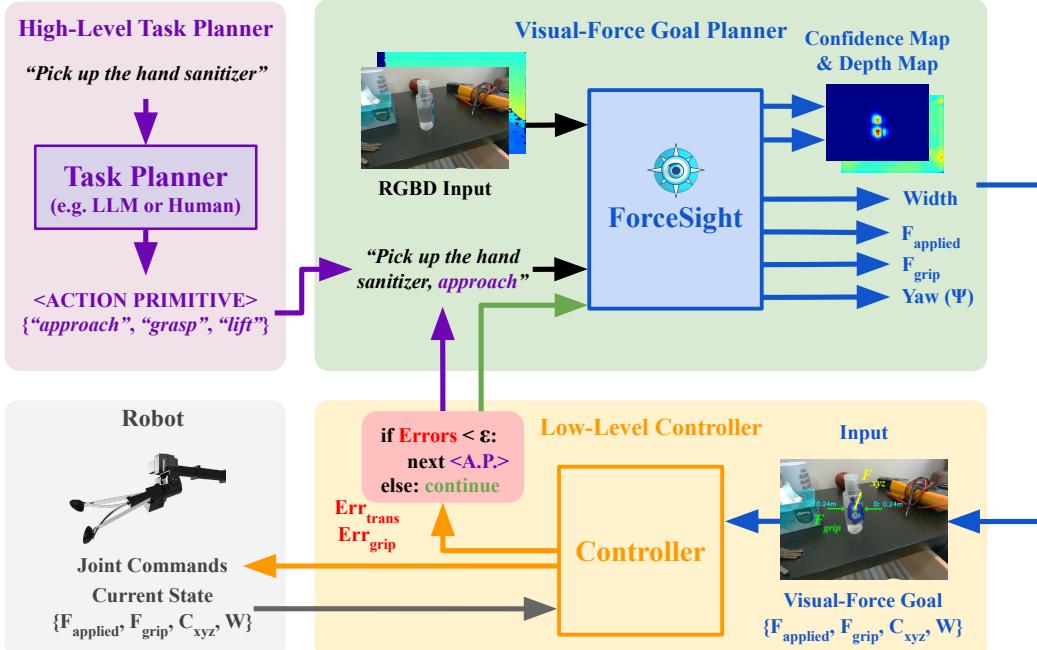


Figure 12: Overall System Architecture.

- 506 The ForceSight system architecture comprises several components that work together to accomplish
 507 a text-conditioned task. It begins with a High-level Task Planner, which takes a text input and
 508 generates a sequence of action primitives representing subgoals. These action primitives, along with
 509 the RGBD input, are then passed to the ForceSight transformer model.
- 510 The ForceSight model processes the input and produces force-based objectives. These objectives
 511 are subsequently fed into the low-level controller, which generates joint motion commands for the
 512 robot to execute the task and its action primitive. This is done by using visual-servo control.
- 513 To determine when to switch to the next action primitive, the low-level controller compares the error
 514 between the current states and visual-force goals with a predefined threshold. If the error is below
 515 the threshold, the low-level controller initiates the switch to the next action primitive.
- 516 This entire process loop operates at a frequency of 8 Hz, allowing for multiple iterations until the
 517 task is completed successfully.

518 H Low-Level Controller

- 519 Our low-level controller receives 3D target contact and force goals located in the camera frame
 520 from ForceSight and executes an action after each image frame observation in order to achieve these
 521 goals. Using an eye-in-hand camera, the low-level controller uses visual-force servoing to move
 522 the gripper closer to the visual-force goal. This approach reduces relative error and is insensitive to
 523 global calibration.
- 524 In our low-level control framework, we divide operations into two main components: Translation
 525 Control and Gripper Control.
- 526 End-Effector Control: Minimizing Euclidean Translation, $\mathcal{E}_{\text{translation}}$ and Applied Force Error,
 527 $\mathcal{E}_{\text{applied force}}$ Our aim here is to minimize both the Euclidean translation and the applied force error
 528 along the XYZ axis:

$$\begin{aligned}\mathcal{E}_{\text{translation}} &= \|C_{\text{predicted}} - C_{\text{current}}\|_2 \\ \mathcal{E}_{\text{yaw}} &= \|\psi_{\text{predicted}} - \psi_{\text{current}}\|_2 \\ \mathcal{E}_{\text{applied force}} &= \|F_{\text{applied, predicted}} - F_{\text{applied, current}}\|_2\end{aligned}\tag{1}$$

- 529 Where $C_{\text{current}} \in \mathbb{R}^3$ is the current translation of the fingertip centroid; $C_{\text{predicted}} \in \mathbb{R}^3$ is the pre-
 530 dicted translation of the fingertip centroid; $F_{\text{applied, current}} \in \mathbb{R}^3$ is vector of the current applied force;
 531 $F_{\text{predicted}} \in \mathbb{R}^3$ is the vector of the predicted applied force.
- 532 Grasping Control: Minimizing Gripper Width, $\mathcal{E}_{\text{width}}$ and Grip Force Error, $\mathcal{E}_{\text{grip force}}$ For this objective,
 533 the goal is to minimize discrepancies in the gripper width and grip force, expressed as:

$$\begin{aligned}\mathcal{E}_{\text{width}} &= \|W_{\text{predicted}} - W_{\text{current}}\|_2 \\ \mathcal{E}_{\text{grip force}} &= \|F_{\text{grip, predicted}} - F_{\text{grip, current}}\|_2\end{aligned}\tag{2}$$

- 534 Where $W_{\text{current}} \in \mathbb{R}$ is the current gripper width; $W_{\text{predicted}} \in \mathbb{R}$ is the predicted gripper width;
 535 $F_{\text{grip, current}} \in \mathbb{R}$ is the current grip force; $F_{\text{grip, predicted}} \in \mathbb{R}$ is the predicted grip force.
- 536 The goal is to minimize the kinematic and force error in all directions in a step-wise manner. To
 537 better incorporate both kinematic and force modalities into the movement error, we define a joint
 538 objective combining both errors. This is expressed as a movement control $M_{\text{end-effector}} \in \mathbb{R}^3$ and
 539 $M_{\text{gripper}} \in \mathbb{R}$ in cartesian space, and then is executed by the controller in a step-wise manner.

$$\begin{aligned}M_{\text{end-effector}} &= \mathcal{E}_{\text{translation}} + \lambda_{\text{applied}} \mathcal{E}_{\text{applied force}} \\ M_{\text{gripper}} &= \mathcal{E}_{\text{width}} + \lambda_{\text{grip}} \mathcal{E}_{\text{grip force}}\end{aligned}\tag{3}$$

540 The movement value determines the necessary movement that the robot should execute in order to
 541 reach the predicted objectives. By combining both kinematic and force objectives, the task can be
 542 executed more delicately, leading to improved performance. This approach is particularly useful in
 543 determining whether the current action primitive has been successfully completed, and whether the
 544 next action primitive should be passed as an input to ForceSight. Results are shown in Table 2.

545 We recognize that there are many options for implementing the low-level controller; for example,
 546 our policy can easily be substituted for a goal-conditioned reinforcement learning policy [38]. We
 547 could also plausibly replace the F/T sensor with a system that estimates these values from motor
 548 current or from vision [39, 17].

549 I Training Details

550 We train ForceSight for 20 epochs using the Adam optimizer [40], corresponding to a total of
 551 500,000 iterations. Our training procedure processes 224x224 RGBD images in batches of eight
 552 with a learning rate of 5e-5.

553 ForceSight’s loss function is as follows:

$$L_A = - \sum_{i,j=0}^{H,W} (\beta * A \log(\hat{A}) + (1 - A) \log(1 - \hat{A})) \quad (4)$$

$$L_D = A * \|D - \hat{D}\|_1 \quad (5)$$

$$L_{F_A} = \|F_A - \hat{F}_A\|_2 \quad (6)$$

$$L_{F_G} = \|F_G - \hat{F}_G\|_2 \quad (7)$$

$$L_W = \|W - \hat{W}\|_2 \quad (8)$$

$$L_\psi = \|\psi - \hat{\psi}\|_2 \quad (9)$$

(10)

554 Where L_A is the confidence map cross-entropy loss, L_D is the masked gripper depth map MAE
 555 loss, L_{F_A} is the MSE of applied force, L_{F_G} is the MSE of grip force, L_W is the MSE for gripper
 556 width, and L_ψ is the MSE for yaw.

557 ForceSight is trained with the weighted sum of the individual losses:

$$L = \lambda_A L_A + \lambda_D L_D + \lambda_{F_A} L_{F_A} + \lambda_{F_G} L_{F_G} + \lambda_W L_W + \lambda_\psi L_\psi \quad (11)$$

558 Balancing the impact of distinct loss components, we assign coefficients as follows: $\lambda_A = 1$, $\lambda_D =$
 559 $5e4$, $\lambda_{F_A} = 0.2$, $\lambda_{F_G} = 0.2$, $\lambda_W = 0.2$, and $\lambda_\psi = 0.2$. In our weighted cross-entropy loss as-
 560 sociated with confidence map prediction (Equation 4), we place a stronger emphasis on the correct
 561 localization of contact points by assigning a β value of 100, emphasizing the importance of correctly
 562 identifying locations of interest.

563 Our ground truths for pixel-wise classification take the form of multi-hot encodings, where circles
 564 with a radius of 10 pixels pinpoint the tool center point’s coordinates. We found that this denser
 565 representation of the ground truth produced rich heatmaps while maintaining good performance.

566 To further improve the robustness of our model and encourage generalization, we apply brightness,
 567 saturation, contrast, and hue augmentation to the RGB channels of the input.

568 We also include a data filtering step that excludes examples where the ground truth lies outside
 569 the camera’s field of view. This process minimizes potential inaccuracies that could arise from
 570 attempting to predict beyond the field of view.

571 Additionally, we find that appending action primitives to the text prompts enhances the model’s per-
 572 formance. This strategy clearly defines the boundaries between keyframes, enhance the transitions
 573 between subgoals. (Table 6).

574 **J Using Large Language Models to Assign Action Primitives**

* GPT-4

You are a component of a robotic planner that takes task descriptions and outputs sequences of action primitives in the form of Python code.

You must select action primitives from the following list:
approach, done, grasp, lift, pull, push, ungrasp

I will provide task descriptions and you will output sequences of action primitives, formatted as lists of strings in Python, e.g. "task_1 = [...] # pick up the apple"

1. "pick up the apple"
2. "pick up the medicine bottle"
3. "pick up the keys"
4. "pick up the paperclip"
5. "pick up the hand sanitizer"
6. "pick up the cup"
7. "place object in the trash"
8. "place object in the hand"
9. "turn off the light switch"
10. "open the drawer"

For each task description, I will provide a sequence of action primitives. Some tasks might be more ambiguous than others (e.g., "turn off the light switch" does not directly involve the action primitives provided), but I'll attempt a best-guess interpretation using the primitives at hand:

```
python Copy code
task_1 = ["approach", "grasp", "lift"] # pick up the apple
task_2 = [ "approach", "grasp", "lift"] # pick up the medicine bottle
task_3 = ["approach", "grasp", "lift"] # pick up the keys
task_4 = ["approach", "grasp", "lift"] # pick up the paperclip
task_5 = [ "approach", "grasp", "lift"] # pick up the hand sanitizer
task_6 = ["approach", "grasp", "lift"] # pick up the cup
task_7 = [ "approach", "ungrasp"] # place object in the trash (assuming the
task_8 = ["approach", "ungrasp"] # place object in the hand (assuming the o
task_9 = ["approach", "grasp", "push", "ungrasp"] # turn off the light switch
task_10 = ["approach", "grasp", "pull", "ungrasp"] # open the drawer
```

Figure 13: We demonstrate how a large language model (GPT-4) could plausibly be used to assign action primitives to task descriptions.

575 In place of the finite state machine that determines which action primitive will be appended to the
576 input prompt, we hypothesize that a large language model could assign action primitives to unseen
577 prompts. We support this hypothesis by prompting GPT-4, a large language model, to perform this
578 operation, and observe promising initial results (Figure 13).