Multi-Contact Task and Motion Planning Guided by Video Demonstration

Kateryna Zorina, David Kovar, Florent Lamiraux, Nicolas Mansard, Justin Carpentier, Josef Sivic, Vladimir Petrik

Multi-Contact Task and Motion Planning Guided by Video Demonstration

Kateryna Zorina[♣] David Kovar[♣] Florent Lamiraux[◇] Nicolas Mansard[◇]
Justin Carpentier[♥] Josef Sivic[♣] Vladimir Petrik[♣]

***Abstract*— This work aims at leveraging instructional video to guide the solving of complex multi-contact task-and-motion planning tasks in robotics. Towards this goal, we propose an extension of the well-established Rapidly-Exploring Random Tree (RRT) planner, which simultaneously grows multiple trees around grasp and release states extracted from the guiding video. Our key novelty lies in combining contact states, and 3D object poses extracted from the guiding video with a traditional planning algorithm that allows us to solve tasks with sequential dependencies, for example, if an object needs to be placed at a specific location to be grasped later. To demonstrate the benefits of the proposed video-guided planning approach, we design a new benchmark with three challenging tasks: (i) 3D re-arrangement of multiple objects between a table and a *shelf*, (ii) multi-contact transfer of an object through a *tunnel*, and (iii) transferring objects using a tray in a similar way a *waiter* transfers dishes. We demonstrate the effectiveness of our planning algorithm on several robots, including the *Franka Emika Panda* and the *KUKA KMR iiwa*.**

I. Introduction

Traditional robot motion planning algorithms seek a collision-free path from a given starting robot configuration to a given goal robot configuration [1]. Despite the large dimensionality of the configuration space, sampling-based motion planning algorithms [2], [3] have shown to be highly effective in practice for solving complex motion planning problems for robots, ranging from six degrees of freedom (DoF) for industrial manipulators to tens of DoFs as for humanoids [4]. Manipulation task-and-motion planning (TAMP) [5] adds an additional complexity to the problem by including movable objects in the state space. This requires the task-and-motion planner to discover the pick-and-place actions that connect the given start and goal robot configurations bringing the manipulated objects from their start poses to their goal poses. The combined need to plan the continuous robot motion together with discrete grasp and release actions makes it very challenging to find a plan for long sequential tasks involving several pick-and-place actions.

To address this challenge, we propose to leverage human demonstrations of the given task provided by an instructional video. Changes of contacts, which are difficult to discover by planning, are extracted from the instructional video together with the 6D poses of *a priori* known objects as visualized in Fig. 1. The extracted information is then used to guide the proposed RRT-like algorithm that simultaneously grows multiple trees around the grasp and release states recognized in the video. To show the benefits of guidance by an instructional video, we design a new benchmark with three challenging multi-contact tasks for manipulation task-and-motion planning that are inspired by real-life problems and are difficult to solve otherwise by state-of-the-art TAMP solvers.

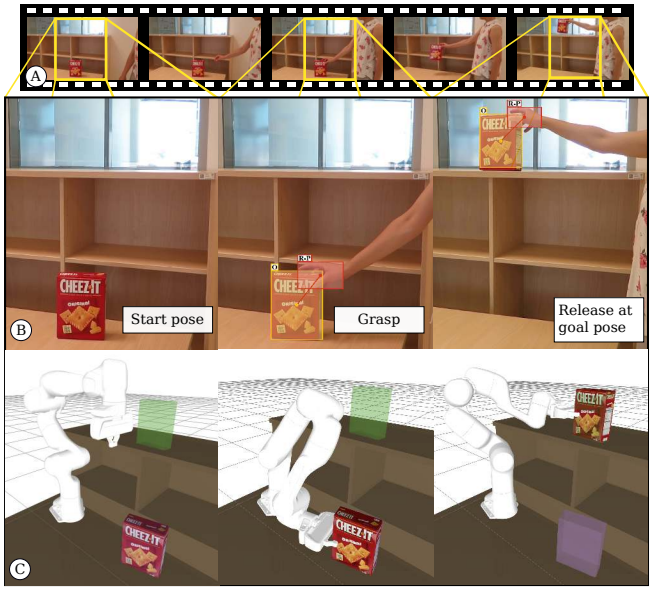

Fig. 1. The proposed planning approach is guided by the demonstration video (A). The video depicts a person manipulating a known object; the *cheez-it* box in this particular example. The video can contain several pick-and-place actions with multiple objects. Here only a short clip with only one object and one action is shown. From the video we recognize (i) the contact states between the human hand and the object, marked by red bounding boxes in (B); and (ii) the object 6D pose (3D translation and 3D rotation w.r.t camera) at the grasp and release contact states, marked in yellow in (B). The robot trajectory planned by the proposed approach is shown in (C). The start and goal object poses in (C) are shown in magenta and green, respectively.

The contributions of this paper are three-fold:

- we propose an RRT-like multi-tree planner that incorporates video demonstration to solve complex multi-contact task-and-motion planning tasks;
- we design a new benchmark with three challenging tasks that we will make publicly available to facilitate progress on this challenging problem;
- we provide experimental results demonstrating that our approach outperforms state-of-the-art TAMP solvers on the proposed benchmark.

We will make the code and data publicly available upon paper acceptance.

♣ CIIRC, Czech Technical University in Prague
◇ LAAS-CNRS, Universite de Toulouse, CNRS, Toulouse
♥ INRIA, Paris

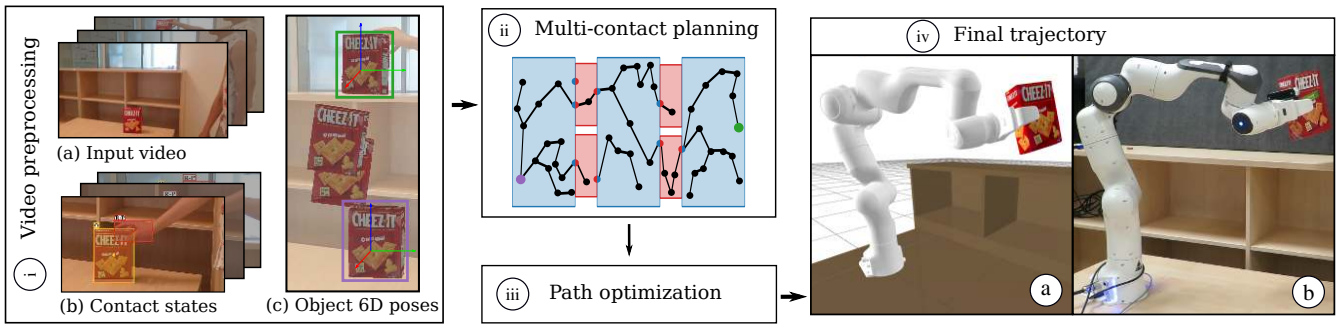

Fig. 2. **Approach overview.** (i) First, we extract contact states and 6D object poses from the input instructional video as described in Sec. III-A. (ii) Next, we grow multiple trees in the admissible configuration space until we find a path between the start and goal configurations. More details on the state space are in Sec. III-B and more details on planning the path in Sec. III-C. (iii) This path is then further shortened by an optimization module, and (iv) executed either in simulation (iv-a) or on a real-world robot (iv-b).

II. Related work

Motion planning. A popular choice for planning the motion of robotic manipulators is to use sampling-based planners inspired by RRT [6], for example, RRT connect [3] that grow trees simultaneously from the start and goal configurations. Our approach is also inspired by RRT but simultaneously grows multiple trees. Using multiple trees for motion planning has been explored in, for example, [7], [8], [9]. In [7], two categories of trees are used: (i) global trees, that are initialized at the start and goal configurations and (ii) local trees, that are initialized from configurations that fail to connect to global trees. Local trees have a higher probability of being constructed in difficult regions such as narrow passages. The trees grow independently until merged with each other. Authors of [8] and [9] explore multi-tree approaches for solving multi-goal planning problems by constructing a tree for every target location. In our approach, the additional trees are sampled at contact state changes that are hard to explore by the planner alone. In our case, the input video demonstration provides the time of the contact state change and the 6D poses of objects for which we sample the 'local' trees. This provides a significant advantage compared to the other sampling methods.

Manipulation task-and-motion planning seeks the robot motion that manipulates objects to achieve their given goal poses. This manipulation is often referred to as object rearrangement planning and was studied by several TAMP solvers [5], [10], [11], [12], [13] with some approaches limiting their applicability to 2D tabletop manipulation [11], [12], [13] where objects are picked and placed only on a given flat surface of a table. Compared to these methods, we solve 3D rearrangement tasks where objects are moved outside the flat surface of the table, for example, onto a shelf. An existing approach not limited to 2D rearrangement planning [10] combines symbolic planning with conditional samplers and has been shown to be effective for multi-contact and cost-sensitive problems. However, it still suffers from the limitations of sampling-based methods. For example, discovdering narrow passages in the configuration space remains a major open challenge. In the proposed work, the input instructional video is used to guide the planner through these narrow passages in the configuration space. Another general TAMP solver is called *Humanoid Path Planner* (HPP) [5] and uses RRT connect [3] to solve TAMP tasks. The planning in HPP is performed in admissible configuration space that is represented by a set of manifolds (denoted *states* in HPP) defined by placement and/or grasp numerical constraints. The movement between the states is achieved through *transitions* that allow *e.g.* to grasp the object at a specific handle. We build on the HPP planner and construct an admissible configuration space based on the input video demonstration.

Guiding planners by demonstration without using video has been explored in several set-ups. For example, several kinesthetic demonstrations obtained by operators guiding the robot manually are used in [14] together with a sampling-based planner to solve household tasks like transferring sugar with a spoon or cleaning a table. Demonstrations in 3D simulation obtained by manipulating a 22 DoFs glove are used in [15] to provide a sequence of actions and to identify parts of the objects that are grasped. The planner is finding a collision-free path for the robot to repeat the sequence of pick-and-place actions. Instead of obtaining demonstrations by kinesthetic guidance or teleoperation by a glove, we propose to extract demonstrations from a video depicting the human performing the given task.

Related to us, video of human demonstration has been used to obtain a sequence of commands for a robot in [16]. It is assumed that a robot controller is available to perform these commands, for example, to grasp or to carry an object, or to pour from the cup. Therefore only the task-space action planning is performed. In our approach, we perform task-and-motion planning and do not rely on given internal controllers to perform the individual manipulation actions. Instead, we rely on a contact recognizer [17] and a 6D pose estimator [18] to extract the relevant guidance for the planner from the video. Such contact and object pose estimators from images have been shown to be robust to various lighting conditions, camera properties, etc. While there are several methods to estimate finger-level grasps of objects [19], [20], robots do not typically grasp objects in the same way as humans and therefore the robotic planner may not benefit from this information. Hence, we focus only on information related to the pose of objects and the contact states, and bypass completely the challenging task of extracting finger-

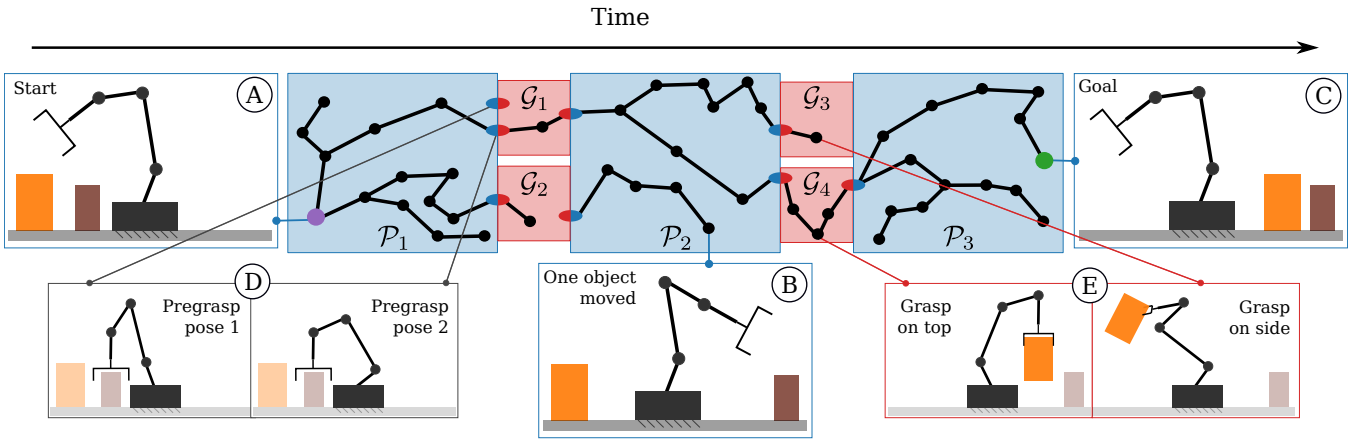

Fig. 3. **The admissible configuration space** is a set of (i) placement states $\mathcal{P}_i$ where objects rest on the contact surfaces at specific poses and (ii) grasp states $\mathcal{G}_j$ where one of the objects is grasped by the robot gripper. To transit to another placement state, *i.e.* to change the pose of one object, robot needs to grasp the object and release it at the new location. Since we define several ways to grasp each object, we can transition through multiple grasp states to build a path between configurations given by two consecutive placement states. Start configuration (A) lies in the first placement state $\mathcal{P}_1$. To achieve the goal, robot needs to move the brown object first, *i.e.* to reach state $\mathcal{P}_2$ (*e.g.* configuration (B)). Finally, robot moves the second object (reach $\mathcal{P}_3$) and moves robot configuration to the given goal (C). The configuration at the transition between the placement state and grasp state is not unique as there are various robot joint values resulting in the same 'pregrasp' poses as shown in (D). Multiple ways of grasping the same object are represented by different grasp states as shown in (E).

level grasping information from the input video.

Learning-based methods. Policy search methods, for example, reinforcement learning, are widely used to solve manipulation tasks [21], [22], [23]. However, these methods usually require substantial training time, and once the policy is learned, it usually does not generalize well to different scenes, for example, if the furniture is moved. To avoid these limitations, we rely on a geometry-based planning approach that is able to re-plan the path if the geometry of the scene changes.

III. Multi-contact planning guided by video demonstration

The proposed method uses a video demonstration to guide the *task & motion* planning algorithm for the given manipulation task. The overall pipeline is visualized in Fig. 2. The input to the proposed approach is the instructional video depicting human manipulating objects and the geometrical description of the scene. The output of the approach is a collision-free path for the robot that solves demonstrated manipulation task.

A. Extracting contact states and object poses from input video

The objective of this stage is to extract the changes of the contact states, *i.e.* when a human grasps or releases an object, and extract the object 6D poses at the times of the contact state changes. The input video demonstration depicts a human manipulating known objects one-by-one from their starting poses towards their goal poses. We focus the proposed method on a single-arm robotic manipulator; therefore, the human demonstrator only moves one object at a time. We apply the hand contact recognizer [17] to obtain a sequence of contact states $c_k \in \{\text{grasp, release, none}\}^N$, where N is the number of movable objects, the contact changes are indexed by $k \in \{1, \ldots, T\}$ with T being the number of changes in contact states. Therefore, the variable c_k contains the contact state for each of the objects in the scene at the time of contact state change k. The contact states of the non-manipulated objects are set to 'none'. The 3D models of objects used in the demonstration are assumed to be known, which is a reasonable assumption in many practical robotics set-ups such as in manufacturing. As a result, the object 6D poses can be estimated from a single frame of the video by a render & compare pose estimator such as CosyPose [18]. The 6D pose is estimated for every movable object in the scene at the time of the contact state change k. The poses of the objects are denoted as $A_k \in SE(3)$, where for each of the objects we estimate the 4×4 homogeneous matrix that describes the pose of the object in the common frame of reference. We use the recognized contact states and object poses to constrain the configuration space in which the planning is performed as described next.

B. The configuration space built from contacts and poses

In case of manipulation task & motion planning, planning is performed in an admissible configuration space, which is defined by a set of numerical constraints that prevent objects from flying in the air or moving without being grasped by the robot. In [5], the space of admissible configurations is represented by a set of states, *i.e.* a set of manifolds, in which the admissible configurations lie. Here a configuration is defined by the position and orientation of objects in the scene as well as the configuration of the robot. The configurations that belong to these states are subject to placement and/or grasp constraints. The placement constraint for an object enforces that the object remains in the same stable pose on the contact surface. A grasp constraint for an object enforces that the object remains static w.r.t. to the gripper frame. For each object, we define several possible handles, *i.e.* several transformations between the robot end-effector and the object. We define *placement states* as those that are

subject to only the placement constraint and *grasp states* as those that are subject to the grasp constraint for one of the objects and placement constraints for the rest of the objects. The transition between the placement and the grasp states represents configurations in which an object is grasped but lies on the contact surface at the same time, *i.e.* constraints from the placement and grasp states are satisfied at the same time. We call such states *neighboring states*.

To simplify the planning, we propose to simplify the admissible configuration space by using the recognized contact states and object poses extracted from the instructional video. For each unique set of static object poses estimated from the video, we construct a placement state. The constructed placement states represent the sequence of object placements observed in the video. The consecutive placement states are *connected* by grasp states as the object needs to be grasped and moved in order to change the placement state, *i.e.* to change the pose of one object. There are multiple handles for each object and therefore there are multiple grasp states, *i.e.* multiple ways to grasp a specific object. An example of such an admissible configuration space is shown in Fig. 3 where three placement states are created: the first placement state $\mathcal{P}_1$ is constrained (given) by the initial poses of objects, the next placement state $\mathcal{P}_2$ is constrained by the starting pose of the orange object and the goal pose of the brown object, and the last placement state $\mathcal{P}_3$ is constrained by the goal poses of both objects. In the example, two handles were defined for each object, representing (i) grasp on top and (ii) grasp on the side. For the path to be feasible, it must contain only the configurations from the admissible space and satisfy additional constraints of the environment, *e.g.* robot configuration must be within the joint limits, and the configuration must be collision-free. We refer to the space in which the configurations satisfy all restrictions as $\mathcal{C}_{\text{free}}$.

C. Planning between contact states

Transitioning between the placement and grasp states, which is necessary to move the objects, remains a challenge for sampling-based planners such as RRT [3] as it needs to sample configurations that satisfy constraints from both neighboring states. To address that issue, we design an extension of RRT that grows multiple trees simultaneously with tree roots sampled at the transitions between placement and grasp states. The overview of the proposed algorithm is shown in Alg. 1. The algorithm is split into two main routines: (i) sampling of a new tree at the transition between the placement and grasp states, and (ii) growing an existing tree at a randomly selected state. We randomly select which of the routines is used in each iteration with a Bernoulli distribution controlled by a parameter $\eta_{\text{sample\_tree}}$. The algorithm stops if both start and goal configurations are connected into a single tree or if we run out of resources (*e.g.*, maximum number of iterations or maximum planning time).

Sample from the state. One of the main capabilities required by the algorithm is sampling from the given state. Similarly to [5], we sample a random configuration from Euclidean space and call a numerical solver [24] iteratively to compute the configuration that satisfies the numerical constraints of the state. To sample from the transition connecting two states, the constraints from both states are merged. However, to assign a unique state to each configuration sampled from the transition, we construct two identical configurations denoted $\boldsymbol{q}_{\text{from}}$ and $\boldsymbol{q}_{\text{to}}$ and assign states to them. The two main routines of Alg. 1, which use the described sampling procedure, are discussed next.

Algorithm 1 Multi-contact RRT guided by demonstration

Require: Contact states c_k, object 6D poses A_k, $\boldsymbol{q}_{\text{start}}$, $\boldsymbol{q}_{\text{goal}}$, $\eta_{\text{sample\_tree}}$, step size δ

$\mathcal{T} \leftarrow \{\text{init\_tree}(\boldsymbol{q}_{\text{start}}), \text{init\_tree}(\boldsymbol{q}_{\text{goal}})\}$ ▷ Existing trees
repeat
  $p \sim \mathcal{U}(0, 1)$
  if $p < \eta_{\text{sample\_tree}}$ **then** ▷ Sampling a new tree
    $k \sim \{1, \ldots, T\}$
    $\boldsymbol{q}_{\text{from}}, \boldsymbol{q}_{\text{to}} \leftarrow \text{sample\_on\_transition}(c_k, A_k)$
    if $\boldsymbol{q}_{\text{from}} \notin C_{\text{free}}$ **or** $\boldsymbol{q}_{\text{to}} \notin C_{\text{free}}$ **then**
      continue
    end if
    $t \leftarrow \text{init\_tree}(\boldsymbol{q}_{\text{from}})$
    $t.\text{add\_edge}(\boldsymbol{q}_{\text{from}}, \boldsymbol{q}_{\text{to}})$
    $\mathcal{T} \leftarrow \mathcal{T} \cup \{t\}$
    $\text{attempt\_link}(\boldsymbol{q}_{\text{from}}, \mathcal{T}, \delta)$ ▷ Alg. 2
    $\text{attempt\_link}(\boldsymbol{q}_{\text{to}}, \mathcal{T}, \delta)$ ▷ Alg. 2
  else ▷ Tree growing
    $\mathcal{S} \leftarrow \text{sample\_state}(\mathcal{T})$
    $\boldsymbol{q}_{\text{rand}} \leftarrow \text{sample\_configuration}(\mathcal{S})$
    $\boldsymbol{q}_{\text{nn}} \leftarrow \text{nearest\_neighbor}(\boldsymbol{q}_{\text{rand}}, \mathcal{S}, \mathcal{T})$
    $\boldsymbol{q}_{\text{step}} \leftarrow \boldsymbol{q}_{nn} + \delta(\boldsymbol{q}_{\text{rand}} - \boldsymbol{q}_{\text{nn}})$
    if $\boldsymbol{q}_{\text{step}} \notin C_{\text{free}}$ **then**
      continue
    end if
    $t \leftarrow \text{get\_tree}(\boldsymbol{q}_{\text{nn}})$
    $t.\text{add\_edge}(\boldsymbol{q}_{\text{nn}}, \boldsymbol{q}_{\text{step}})$
    $\text{attempt\_link}(\boldsymbol{q}_{\text{step}}, \mathcal{T}, \delta)$ ▷ Alg. 2
  end if
until $\text{get\_tree}(\boldsymbol{q}_{\text{start}}) = \text{get\_tree}(\boldsymbol{q}_{\text{goal}})$ or out of resources

Sampling a new tree at the transition between the placement and grasp states is performed as follows. First, we sample a pair of configurations $(\boldsymbol{q}_{\text{from}}, \boldsymbol{q}_{\text{to}})$ that satisfy the constraints of the randomly selected transition. If both configurations also satisfy the constraints of the environment (*i.e.* respect the joint limits and are collision-free), we create a new tree containing a root ($\boldsymbol{q}_{\text{from}}$) and a leaf ($\boldsymbol{q}_{\text{to}}$). We attempt to link both the created configurations to the existing trees in their corresponding states.

Tree growing in a randomly selected state is performed as follows. This routine consists of sampling a new configuration in admissible configuration space and extending the tree in the direction of the sampled configuration. We randomly choose a state $\mathcal{S}$ and sample the random configuration $\boldsymbol{q}_{\text{rand}}$ from it. The nearest neighbor of the sampled configuration $\boldsymbol{q}_{\text{nn}}$ is found such that $\boldsymbol{q}_{\text{nn}}$ also lies in the state $\mathcal{S}$. A new configuration $\boldsymbol{q}_{\text{step}}$ is computed along the

segment between $\boldsymbol{q}_{\text{nn}}$ and $\boldsymbol{q}_{\text{rand}}$ in the manually defined step-size distance δ from $\boldsymbol{q}_{\text{nn}}$. If $\boldsymbol{q}_{\text{step}}$ is collision-free, we add it to the tree. Finally, we attempt to link the new configuration to existing trees that have nodes in the same state.

Attempt to link function is used by Alg. 1 to connect given configuration $\boldsymbol{q}$ with the existing trees. The function is shown in Alg. 2. It is searching for a linear path between the configuration $\boldsymbol{q}$ and other trees that have nodes in the same state. For each tree, the nearest neighbor $\boldsymbol{q}_{\text{nn}}$ is found. Then, configurations along the segment from $\boldsymbol{q}_{\text{nn}}$ to $\boldsymbol{q}$ are added to the tree unless a collision is observed. If the entire path is collision-free, we merge the trees that contain the configurations $\boldsymbol{q}_{\text{nn}}$ and $\boldsymbol{q}$.

Algorithm 2 Attempt to link function connects a given configuration to other trees

Require: configuration $\boldsymbol{q}$, set of trees $\mathcal{T}$, step size δ,
$\mathcal{S} \leftarrow$ get_state($\boldsymbol{q}$)
for $t \in \mathcal{T} \setminus$ get_tree($\boldsymbol{q}$) **do**
  if t has nodes in $\mathcal{S}$ **then**
    $\boldsymbol{q}_{\text{nn}} \leftarrow$ nearest_neighbor($\boldsymbol{q}$, $\mathcal{S}$, $\{t\}$)
    $\boldsymbol{q}_{\text{parent}} \leftarrow \boldsymbol{q}_{\text{nn}}$
    for $\boldsymbol{q}_{\text{step}} \in \{\boldsymbol{q}_{\text{nn}}, \boldsymbol{q}_{\text{nn}} + \delta, \ldots, \boldsymbol{q}_{\text{nn}} + n\delta, \boldsymbol{q}\}$ **do**
      if $\boldsymbol{q}_{\text{step}} \notin C_{\text{free}}$ **then**
        break
      end if
      t.add_edge($\boldsymbol{q}_{\text{step}}$, $\boldsymbol{q}_{\text{parent}}$)
      $\boldsymbol{q}_{\text{parent}} \leftarrow \boldsymbol{q}_{\text{step}}$
    end for
    if $\boldsymbol{q}_{\text{step}} = \boldsymbol{q}$ **then**
      merge t and tree containing $\boldsymbol{q}$
    end if
  end if
end for

D. Path optimization

The proposed multi-contact RRT produces a feasible path in the admissible configuration space that achieves the goal. However, this path often contains redundant motion, for example, approaching an object several times before grasping it. To further optimize the path, we apply an approach similar to [25]: we select two random configurations $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ that lie on the path and belong to the same state. We try to build a collision-free segment between them. If such a segment exists, we replace the portion of the path between $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ by this segment.

IV. Experiments

In this section we first describe the new benchmark for multi-contact task & motion planning and then give the details of our evaluation including a discussion of the benefits and limitations of our approach.

A. Benchmark for multi-contact task & motion planning

We designed a new benchmark consisting of three challenging tasks that involve multiple contact changes, *i.e.* object pick-and-place actions. The objects used for manipulation were selected from the YCBV dataset [26] for which the 6D pose estimator [18] is available. The camera used for the video demonstration recording was calibrated both intrinsically and extrinsically with respect to the environment. This allows us to express object poses and furniture poses in a common frame of reference. Multiple robots with varying number of degrees of freedom are tested in each task, including the Panda robot and the Kuka IIWA robot mounted on a mobile platform. For the purpose of benchmarking, we sample the attachment pose of the non-mobile robots to the desk randomly, such that all pick-and-place poses in the demonstration are reachable. For real scenarios, the robot-to-desk attachment is given by the environment. Next, we provide the description of the tasks. Please see the supplementary video for their visualization.

The shelf task is composed of a table, a shelf and a varying set of objects that the robotic manipulator is supposed to arrange, *i.e.* to move the objects to the predefined poses on the table or on the shelf. The complexity of the task is controlled by the number of objects that the robot should arrange. This task is challenging for state-of-the-art planners because it requires moving multiple objects in a single planning task.

The tunnel task consists of a tunnel and a single object that should be transferred through the tunnel. The tunnel is thin enough so that the robot can place an object inside the tunnel on one side and pick it up from the other side. The challenge lies in the narrow passage in the admissible configuration space that needs to be discovered by the planner.

The waiter task simulates the job of waiter, in which a set of objects needs to be transferred from one location to another distant location. Waiters use a tray to minimize the walked distance. In our simulation, a mobile robot is equipped with a tray-like space that it can use for transferring objects. Discovering the tray in the planning state-space is non-trivial, which makes this task challenging for the planners that do not utilize demonstrations.

B. Evaluation

Baselines. We compare the proposed approach with the following baselines: (i) RRT-connect implemented in HPP [5] without and (ii) with path optimization, (iii) PDDLStreams method [10], and (iv) the proposed approach without and (v) with the proposed path optimization. For PDDLStreams we set the goal requirement to match the final object poses for every object. The object grasp definitions are set to be consistent with the proposed approach.

Metrics. To compare the proposed approach with the baselines, we solve the planning problem on the same machine and use the following metrics: (i) the time to find a solution measured in seconds, (ii) the number of object grasps and releases, *i.e.*, the number of contacts, and (iii) the length of the planned path. The length of the planned path is evaluated as the sum of Euclidean distances computed between the consecutive robot configurations of the planned path.

Results are shown in Fig. 4. We set the time limit to 60s for all tasks except the waiter task where the limit was set to 300s. Planning was repeated 10 times and the success rate is

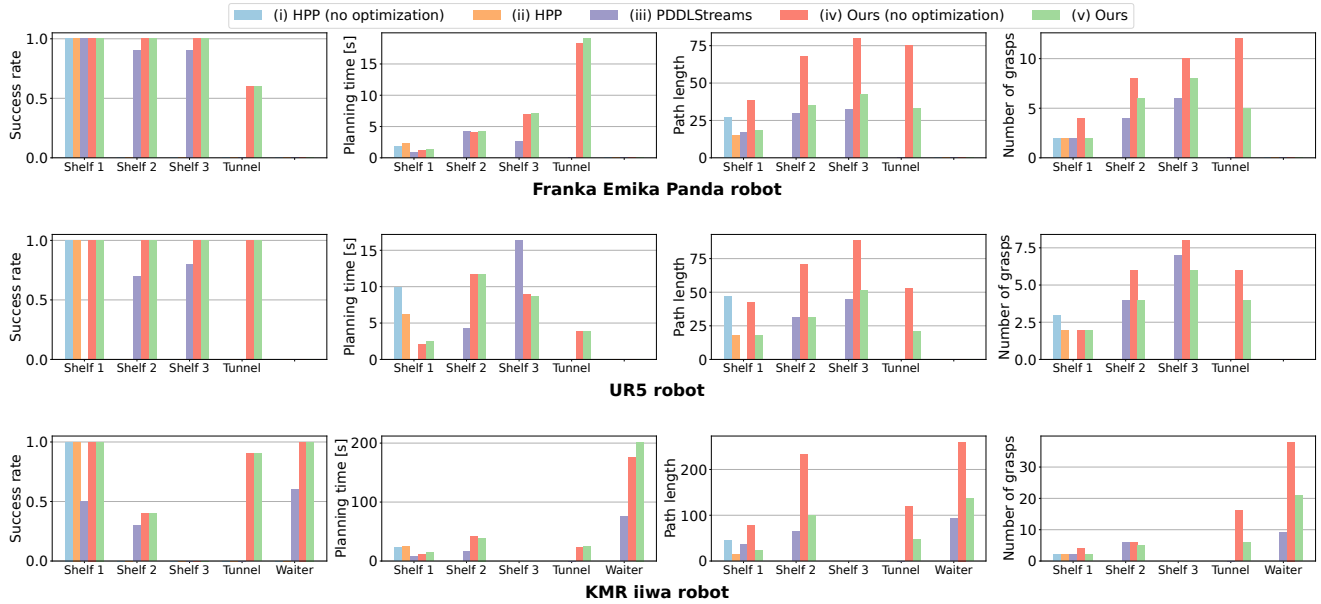

Fig. 4. **Results** reported for different robots (rows) and different metrics (columns). We report (from left): the success rate, the planning time [s], the path length, and the number of grasps. For the success rate, the higher number the better; for the other metrics lower numbers are better. The plots show the comparison of our method with the following baselines: (i) RRT-connect implemented in HPP [5] without and (ii) with the path optimization, (iii) PDDLStreams planner [10], and (iv) our proposed approach without and (v) with the proposed path optimization. Each graph shows results for five different tasks. "Shelf 1" - "Shelf 3" correspond to the Shelf tasks with 1-3 objects. "Tunnel" corresponds to the tunnel task and "Waiter" corresponds to the waiter task. Please note that the Waiter task is only performed with the KMR iiwa mobile robot.

reported in the first column of Fig. 4. For the successful runs, we measure the aforementioned metrics and report the average. Different columns in Fig. 4 show the different reported metrics. Different rows in Fig. 4 correspond to experiments with different robots (from top): (i) 7 DoFs Franka Emika Panda robot, (ii) 6 DoFs UR5 robot, and (iii) 7 DoFs KUKA iiwa arm on 3 DoFs mobile platform (*i.e.* KMR iiwa robot). We sample the robot base poses such that all objects are reachable in both the start and the goal poses. The same robot base pose was used for all the methods. Next, we discuss the results for each task separately.

For the shelf task, our method outperforms the state-of-the-art solvers [10], [5] based on the success rate. With Panda and UR5 robots we solve all shelf tasks with 100 % success rate. However, for the successful runs, the PDDLStream planner [10] arrives at a solution faster and the final planned path is shorter on average. HPP [5] fails to solve shelf tasks for more than one object in the given time limit. For the KMR robot, all methods fail to solve the task with three objects, most likely due to the large base of the robot colliding with the environment. To conclude, the main benefit of guiding the planner by video for the shelf task lies in the success rate, *i.e.* the guidance by video allows us to solve the given task reliably.

For the tunnel task, the state-of-the-art planners fail to solve the task due to the difficulty of discovering the narrow passage by the sampling-based planning methods. The proposed approach takes advantage of the available demonstration to discover the narrow passage and manages to plan a path for all of the tested robots.

For the waiter task, we achive a higher success rate than the PDDLStream planner [10], however, it takes more time to arrive at a solution. The PDDLStream planner ignores the tray and moves each object one by one, which results in a lower number of grasps.

Limitations and assumptions. Our approach has several assumptions and limitations. We assume that the demonstration consists of a sequence of pick-and-place motions (without pushing or sliding), where we manipulate one object at a time. Start and goal object poses should be reachable by the robot and suitable for grasp. For example, objects should not be too close to each other, otherwise the planning will fail as the gripper is not able to grasp the object. For the waiter task, we assume that objects are manipulated only when the moving platform is stable. For accurate processing of the video demonstration, objects and hands should be well visible at the contact events. Significant occlusions can still pose a challenge for the current approach.

V. Conclusions

We present an approach for guiding task and motion planning by a video demonstration. We show that contact states and 6D object poses extracted from the video help building the tree in the configuration space of the planner much more efficiently. Our evaluation on our newly built task & motion planning benchmark shows that the video demonstration improves the success rate of planning and is especially effective in environments with narrow passages, which present a major challenge for current sampling based planners. This work makes a step towards large-scale learning of task & motion plans from Internet's instructional videos.

REFERENCES

[1] K. M. Lynch and F. C. Park, *Modern robotics*. Cambridge University Press, 2017.

[2] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for fast path planning in high dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[3] J. Kuffner and S. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings of 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, pp. 995–1001 vol.2, 2000.

[4] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion planning for humanoid robots," in *Robotics research. The eleventh international symposium*, pp. 365–374, Springer, 2005.

[5] F. Lamiraux and J. Mirabel, "Prehensile manipulation planning: Modeling, algorithms and implementation," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2370–2388, 2022.

[6] S. M. LaValle *et al.*, "Rapidly-exploring random trees: A new tool for path planning," *The annual research report*, 1998.

[7] M. Strandberg, "Augmenting rrt-planners with local trees," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 4, pp. 3258–3262 Vol.4, 2004.

[8] J. Janoš, V. Vonásek, and R. Pěnička, "Multi-goal path planning using multiple random trees," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4201–4208, 2021.

[9] D. Devaurs, T. Siméon, and J. Cortés, "A multi-tree extension of the transition-based rrt: Application to ordering-and-pathfinding problems in continuous cost spaces," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2991–2996, 2014.

[10] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling, "Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning," *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 30, pp. 440–448, Jun. 2020.

[11] A. Krontiris and K. E. Bekris, "Dealing with difficult instances of object rearrangement.," in *Robotics: Science and Systems*, vol. 1123, 2015.

[12] H. Shuai, N. Stiffler, A. Krontiris, K. E. Bekris, and J. Yu, "High-quality tabletop rearrangement with overhand grasps: Hardness results and fast methods," in *Robotics: Science and Systems (RSS)*, (Cambridge, MA), 07/2017 2017.

[13] Y. Labbé, S. Zagoruyko, I. Kalevatykh, I. Laptev, J. Carpentier, M. Aubry, and J. Sivic, "Monte-carlo tree search for efficient visually guided rearrangement planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3715–3722, 2020.

[14] G. Ye and R. Alterovitz, "Guided motion planning," in *Robotics research*, pp. 291–307, Springer, 2017.

[15] J. Aleotti and S. Caselli, "Part-based robot grasp planning from human demonstration," in *2011 IEEE International Conference on Robotics and Automation*, pp. 4554–4560, 2011.

[16] A. Nguyen, D. Kanoulas, L. Muratore, D. G. Caldwell, and N. G. Tsagarakis, "Translating videos to commands for robotic manipulation with deep recurrent neural networks," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3782–3788, IEEE, 2018.

[17] D. Shan, J. Geng, M. Shu, and D. F. Fouhey, "Understanding human hands in contact at internet scale," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[18] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "Cosypose: Consistent multi-view multi-object 6d pose estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)* (A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, eds.), (Cham), pp. 574–591, Springer International Publishing, 2020.

[19] Z. Cao, I. Radosavovic, A. Kanazawa, and J. Malik, "Reconstructing hand-object interactions in the wild," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12417–12426, October 2021.

[20] Y. Hasson, G. Varol, D. Tzionas, I. Kalevatykh, M. J. Black, I. Laptev, and C. Schmid, "Learning joint reconstruction of hands and manipulated objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[21] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[22] M. P. Deisenroth, C. E. Rasmussen, and D. Fox, "Learning to control a low-cost manipulator using data-efficient reinforcement learning," *Robotics: Science and Systems VII*, vol. 7, pp. 57–64, 2011.

[23] P. Ennen, P. Bresenitz, R. Vossen, and F. Hees, "Learning robust manipulation skills with guided policy search via generative motor reflexes," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 7851–7857, IEEE, 2019.

[24] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York, NY, USA: Springer, 2e ed., 2006.

[25] G. Sánchez and J.-C. Latombe, "On delaying collision checking in prm planning: Application to multi-robot coordination," *The International Journal of Robotics Research*, vol. 21, no. 1, pp. 5–26, 2002.

[26] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *Robotics: Science and Systems (RSS)*, 2018.