Knowledge-oriented Physics-based Motion Planning for Grasping under Uncertainty

Muhayyuddin, Aliakbar Akbari and Jan Rosell*

Institute of Industrial and Control Engineering,
Universitat Politècnica de Catalunya, Barcelona, Spain,
{muhayyuddin.gillani,aliakbar.akbari,jan.rosell}@upc.edu

Abstract. Grasping an object in unstructured and uncertain environments is a challenging task, particularly when a collision-free trajectory does not exits. High-level knowledge and reasoning processes, as well as the allowing of interaction between objects, can enhance the planning efficiency in such environments. In this direction, this study proposes a knowledge-oriented physics-based motion planning approach for a hand-arm system that uses a high-level knowledge-based reasoning to partition the workspace into regions to both guide the planner and reason about the result of the dynamical interactions between rigid bodies. The proposed planner is a kinodynamic RRT that uses a region-biased state sampling strategy and a smart validity checker that takes into account uncertainty in the pose of the objects. Complex dynamical interactions along with possible physics-based constraints such as friction and gravity are handled by a physics engine that is used as the RRT state propagator. The proposal is validated for different scenarios in simulation and in a real environment using a 7-degree-of-freedom KUKA Lightweight robot equipped with a two-finger gripper. The results show a significant improvement in the success rate of the execution of the computed plan in the presence of object pose uncertainty.

1 Introduction

Recent advancements in robotics allow robots to perform challenging tasks in complex environments. Most of the manipulation tasks, such as setting the table, require the ability to execute stable grasps through sequences of grasping actions. Grasp planning has been studied extensively since the past few decades [1–3]. The traditional approaches for grasping an object are *object-centric*. These approaches determine the contact points on the target object in such a way that, if fingers are placed on these points, stable grasps are guaranteed. Afterward, a motion planner (such as PRM [4] or RRT [5]) is launched to determine the collision-free trajectory to move the hand towards the grasping pose. It is desirable for stable grasps that fingers make contact with the object simultaneously and avoiding collisions with other objects in the environment. Otherwise, the target object may be displaced from its location, resulting in an unsuccessful grasp.

* This work was partially supported by the Spanish Government through the projects DPI2013-40882-P and DPI2016-80077-R.

These approaches work well in a structured or semi-structured environment where objects are not too close to each other. However, unstructured and uncertain environments (generally referred as clutter) pose serious challenges, especially when a collision-free trajectory does not exist. Grasping in such environments requires a rich semantic knowledge of the scene and the allowing of robot-object and object-object interactions. That is, to clear the path towards the target object, the robot must have the ability to reason about the consequences of each complex dynamical interaction in order to avoid objects falling from the table/shelf or displacing the target object, thus preventing the successful execution of the task.

This paper aims to contribute in this line by proposing a knowledge-oriented physics-based motion planning approach for grasping in the clutter. The approach also handles the uncertainty in the pose of the objects, which is a key issue to be considered when dynamic interactions occur, and that is important to be taken into account during planning. In this sense, for instance, the work in [6] handled the uncertainty caused by the robot-object interactions (i.e. no object-object interactions were allowed), by analysing them considering linear motions of the gripper and the quasi static assumption (i.e. the interaction forces are small enough to avoid the effect of inertial forces). The proposal presented here is based on a randomized kinodynamic motion planner with a dynamic engine as state propagator that allows to consider any type of motion (not only rectilinear motions of the gripper) and any type of interaction (robot-object and object-object). The planner includes a high-level knowledge about the scene, stored in the form of ontologies, that is used to provide a semantic description of the scene and information on how the robot may interact with the obstacles. Moreover, a smart state-validity checker is introduced that evaluates the post-effect of dynamic interactions and computes a confidence index that provides the belief about the robustness of the selected controls in the presence of uncertainty.

The rest of the paper is structured as follows. First, Sec. 2 introduces some related work and Sec. 3 formulates the problem. Then, the proposed approach is explained in Sec. 4, that details the reasoning process, the uncertainty handling and the sampling-based kinodynamic motion planner. The implementation and the simulation setup is sketched in Sec. 5, together with the results and discussion. Finally, the conclusions are summarized in Sec. 6.

2 Related Work

The approaches for grasping in cluttered environments can be categorized into reconfiguration planning and physics-based grasp planning. Reconfiguration planning approaches clear the way toward the target object by moving the objects away [7–9]. These approaches are still object-centric; in each action (either push or grasp) the focus is on moving a particular object while avoiding the interaction with the rest of the environment. These approaches may not work well in complex environments. On the other hand, physics-based grasp planning are *clutter-centric*; they allow the robot to make contact with the objects and manipulate them in a controlled way for clearing the path towards the target object.

These approaches, that have emerged recently, incorporate the physics-based simulation to evaluate the effect of the dynamic interactions between the robot and the objects in the environment. In this line, some works like [6, 10] perform the physics-based simulation using push mechanics considering the quasi-static assumption. These approaches sample straight line motions for the hand, using precomputed interactions of the robot with each object in an isolated environment (no object-object interactions are considered). The straight line motion that ends with a successful grasp is selected as a solution. Finally, the inverse kinematics is solved to move the arm on the computed path. In a similar line [11] proposes a grasp trajectory optimization approach for the hand-arm system. Rather than the pre-determining quasi-static interactions it uses a dynamic engine (Bullet Physics Library [12]). This approach also plans straight line motions.

The need to consider uncertainty in motion planning arises by the imprecise knowledge of the robot dynamic model and of the initial state, that may condition the success of the task execution. In this line, some approaches, such as [13–15] introduce a stochastic parameter (with zero mean Gaussian distribution) in the robot model that is used for the step propagation of an RRT-based algorithm, as well as a Gaussian distribution around the measured initial state. Other approaches focus on the uncertainty in dynamic environments [16], or in the environment parameters like friction [17]. It has to be noted that the focus of all the above mentioned approaches is to compute collision-free paths and therefore no interactions are considered. On the contrary, the approach in [6] considers object pose uncertainty to compute the region formed by the set of target object poses that result into stable grasps, and uses it to plan linear motions to grasp the target object, allowing quasi-static interactions of the end-effector with the objects.

To make planning more efficient, the use of knowledge to reason on manipulation actions and enhance physics-based motion planning was proposed in [18] for mobile robots, where it was coded as ontologies that stored a high-level description of the scene (in terms of properties of the robot, the objects and the manipulation constraints. Following this approach the present proposal extends it by considering robot manipulators, the reasoning on the effects of dynamic interactions (both robot-object and object-object interactions), and the presence of uncertainty.

3 Problem Formulation

Consider a hand-arm trajectory planning problem to grasp a target object in an uncertain and unstructured environment. The environment E is composed of a robot $\mathcal{R}$ and a set of objects $\mathcal{O}$ lying on an horizontal flat surface. The robot consists of a set of n links described as $\{\mathcal{L}_1 \dots \mathcal{L}_n\}$. Let $\mathcal{X}$ represent the state space of the robot, it is a differential manifold. The state $x \in \mathcal{X}$ of the robot at any time t can be represented as $x(t) = \{\mathcal{L}_1(\boldsymbol{q}_1, \dot{\boldsymbol{q}}_1), \dots, \mathcal{L}_n(\boldsymbol{q}_n, \dot{\boldsymbol{q}}_n)\}$, where $\boldsymbol{q}_i$ and $\dot{\boldsymbol{q}}_i$ represent the angle and the angular velocity of the i-th joint, respectively. The objects in the environment are classified into a set of fixed objects $\mathcal{O}^f$ and a set of manipulatable objects $\mathcal{O}^m$, that includes the target object $\mathcal{O}^m_{\text{target}}$. The robot is only allowed to interact with the manipulatable objects. Then, objects are represented as $\mathcal{O} = \{\mathcal{O}^m_1 \dots \mathcal{O}^m_j, \mathcal{O}^f_1 \dots \mathcal{O}^f_k\}$ where j and k

represent the number of manipulatable and fixed objects, respectively. Let $\mathcal{S}$ be the state space of the objects in the environment. Then, at any time t, the state of the objects can be parametrized as $s(t)$=$\{s_1, \ldots, s_{j+k}\}$ where $s_i \in \mathcal{S}$ has a component p_i representing the pose, and another one v_i representing the linear and angular velocities of each object at time t. The state of the environment is then described as $E = \mathcal{X} \times \mathcal{S}$, and the dynamic state propagator $\boldsymbol{f}$ is defined as;

$$\boldsymbol{f} : E_i \times \mathcal{U} \longrightarrow E_{i+1} \tag{1}$$

Where, $\mathcal{U}$ is the control space containing all possible controls that can be applied to the system. In this work, controls will be represented in terms of joint velocities.

Consider a situation where the path to reach the target object is blocked with one or several objects in such a way that a collision-free trajectory from the start to a goal configuration does not exist. Then, the objective is to plan the grasp approach trajectory by computing the sequence of controls and corresponding duration in such a way that if sequentially applied to the robot (in the presence of object pose uncertainty), it moves the robot from the start to the (pre-grasping) goal state.

Uncertainty in the knowledge of the actual initial poses due to the sensory noise will be considered. The pose uncertainty region will be modelled as $\boldsymbol{U}_{\mathrm{E}_{\mathrm{init}}} = \mathcal{N}(\mathbf{p}^{\mathrm{init}}, \boldsymbol{\nu}^{\mathrm{init}})$, i.e. a multivariate Gaussian distribution with mean $\mathbf{p}^{\mathrm{init}} = (\mathbf{p}_1^{\mathrm{init}}, \ldots, \mathbf{p}_{j+k}^{\mathrm{init}})$, the measured initial pose, and variance $\boldsymbol{\nu}^{\mathrm{init}} = (\nu_1^{\mathrm{init}}, \ldots, \nu_{j+k}^{\mathrm{init}})$. The uncertainty will be propagated to future states as a result of robot-object and object-object interactions. Then, the controls to be selected will be those that, when applied to any actual pose within the pose uncertainty region, are robust enough to bring the system to a new valid state, with a probability higher than a given threshold.

4 The Proposed Approach

The proposed approach works in two main phases that are the pre-processing phase and the planning phase. The former involves a knowledge-based representation of the environment to provide the semantic description of the scene and a reasoning process that partitions the workspace into regions, define manipulation constraints for the objects, and provides the detailed insight about the task. The latter consists of a sampling-based kinodynamic motion planner that uses the high-level description of the problem (computed in the pre-processing phase) to plan efficiently. Moreover, it proposes a strategy to cope with object pose uncertainty. The integration of these two phases enables the robot to plan robustly in complex environments.

4.1 Reasoning Process

To model and represent semantic knowledge, an ontology-based approach is used. Ontologies can mainly collect and categorize different sort of knowledge within various classes embracing a collection of objects, individuals being instances of classes. They can be encoded with the Ontology Web Language (OWL) [19] to allow the sharing of the knowledge over the world wide web. Accordingly, knowledge regarding the robot as well as obstacles properties is stored in two classes using OWL (Fig. 1):

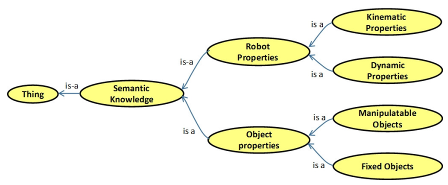

Fig. 1: OWL semantic knowledge taxonomy.

Algorithm 1 ReasoningProcess($\mathcal{K}_\text{s}$, $\mathcal{O}^m_\text{target}$, E_init)

```
1: $\mathcal{K}_\text{m} \leftarrow \emptyset$
2: $\mathit{TRgn}$ $\leftarrow$compute_target_region($\mathcal{O}^m_\text{target}$, $E_\text{init}$)
3: $\{\mathcal{O}^m_{\mathit{TRgn}}\}$$\leftarrow$objects_in_target_region($\mathit{TRgn}$)
4: $\{\mathit{mRgn}\}$$\leftarrow$compute_manipulation_regions($\mathcal{K}_\text{s}$)
5: **return** $\mathcal{K}_\text{m}$.add($\mathit{TRgn}$, $\{\mathcal{O}^m_{\mathit{TRgn}}\}$, $\{\mathit{mRgn}\}$)
```

- *Robot properties*: It includes the robot kinematic properties (such as joint limits and collision model) and dynamic properties (such as masses and bounds on forces and velocities), as well as the properties of the gripper.
- *Obstacles Properties*: It involves properties of obstacles such as the poses of fixed and manipulatable objects, their masses, friction coefficients, and manipulation constraints (represented in terms of manipulation regions).

The reasoning process, summarized in Algorithm 1, is performed over this semantic knowledge, called $\mathcal{K}_\text{s}$, in order to specify the *target region* as well as the *manipulation region* for each of the manipulatable objects. The target region is determined as a box region around $\mathcal{O}^m_\text{target}$, and it is considered as the part of the workspace where efforts to find a robust plan are focused. The manipulation regions (*mRgn*) are defined as the regions around the manipulatable objects from where the robot can interact with the obstacles (in this respect, for instance, these regions will be located below the center of mass for tall and thin objects). This spatial reasoning process is carried out over the knowledge ontologies using the Prolog language, with the following predicates:

- *compute_target_region:* Given $\mathcal{O}^m_\text{target}$ and E_init, it computes the location of the target object and applies spatial reasoning to compute the target region (*TRgn*). It will be modeled as a box centered at the target object with the size of the sides depending on the robot and the objects.
- *objects_in_target_region:* Given a *TRgn*, it returns the set $\mathcal{O}^m_{\mathit{TRgn}}$ of manipulatable objects it contains (including $\mathcal{O}^m_\text{target}$). Uncertainty will be considered only for $\mathcal{O}^m_{\mathit{TRgn}}$.
- *compute_manipulation_region:* It takes the semantic knowledge $\mathcal{K}_\text{s}$ as input and defines the manipulation regions based on the properties of the objects.

The output of the reasoning process is encapsulated as the manipulation knowledge $\mathcal{K}_\text{m}$ that will be used by the motion planner for computing the plan.

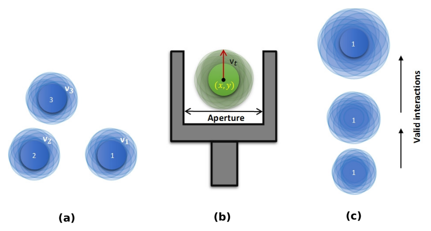

Fig. 2: (a) Initial pose uncertainty of three objects; (b) Pose uncertainty of the target object assumed to fit within the gripper aperture; (c) Enlarging of the pose uncertainty of an object being displaced due to the results of interactions.

4.2 Handling Pose Uncertainty

This section describes: *1)* the evaluation of the outcomes of dynamic interactions done by the validity checker taking into account uncertainty, *2)* the strategy to define the robustness of controls according to a set of possible resultant states, and *3)* the propagation of the uncertainty into the future states as a result of the dynamic interactions.

The initial pose uncertainty $\boldsymbol{U}_{\mathrm{E_{init}}}$ is propagated due to dynamic interactions, i.e. the mean and the deviation vary due to the objects that are moved due to interactions. As an example, Fig. 2-a shows the initial pose uncertainty of three objects (as the projection of $\boldsymbol{U}_{\mathrm{E_{init}}}$ onto the corresponding xy-planes), Fig. 2-c the enlarging of the pose uncertainty as the object is displaced due to an interaction, and Fig. 2-b the pose uncertainty of the target object (it is assumed that the gripper aperture is big enough to envelope it, and this will always hold since no interaction with the target object are permitted).

State validity checker The proposed state-validity checker allows the collision with manipulatable objects, while rejects the collision with fixed objects. The result of an interaction with a manipulatable object will be valid, however, only if the interaction takes place when the robot gripper is located at the object manipulation region, the velocity at the contact is below a given threshold (to prevent unwanted large motions of the objects), and no object collide with the target object or falls from the table.

Control Evaluation The effects of applying a given control when interactions occur depend on many factors, like friction, pressure distributions under the object surface, momentum and inertial effects, as well as on the actual pose of the objects (within the uncertainty region) when the control is applied. Therefore, the use of the state propagator dynamics engine is proposed to evaluate its eligibility when applied to a given state E_{near}. The procedure, sketched in Algorithm 2, relies on applying the control (during a given time duration) from a set of n states sampled from the uncertainty region associated to E_{near}, and evaluating the validity of the resulting states. The probability of obtaining a valid resultant state is estimated by the ratio of valid resultant states w.r.t. the total, and is called confidence index. The main functions used in Algorithm 2 are:

- *SampleState:* Varies the poses of the objects of a given nominal state E_{near} by sampling them from the corresponding pose uncertainty region $\boldsymbol{U}_{\mathrm{E_{near}}} = \mathcal{N}(\mathbf{p}^{\text{near}}, \boldsymbol{\nu}^{\text{near}})$, and returns the resultant state, called E_{trial}.

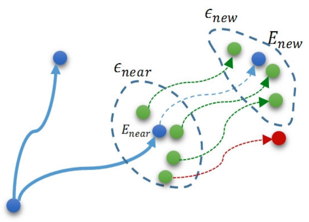

Fig. 3: The robustness evaluation of the selected controls, n controls will be randomly sampled from $\mathcal{E}_{\text{near}}$. The valid resultant states are represented with green color whereas invalid with the red.

Algorithm 2 ConfidenceIndex($\boldsymbol{U}_{\text{E}_{\text{near}}}, u_{\Delta t}, \mathcal{K}_{\text{m}}$)

```
1: $\mathcal{E}_{\text{near}} \leftarrow \emptyset$, $\mathcal{E}_{\text{new}} \leftarrow \emptyset$, validstatecounter $= 0$
2: for $i = 0$ to $n$ do
3:     $E^{\text{trial}}_{\text{near}} \leftarrow$ SampleState($\boldsymbol{U}_{\text{E}_{\text{near}}}$)
4:     $E^{\text{trial}}_{\text{new}} \leftarrow$ Propagate($E^{\text{trial}}_{\text{near}}, u_{\Delta t}$)
5:     if StateValidityChecker($E^{\text{new}}_{\text{trial}}, \mathcal{K}_{\text{m}}$) then
6:         validstatecounter $=$ validstatecounter $+ 1$
7:         $\mathcal{E}_{\text{near}}$.Add($E^{\text{trial}}_{\text{near}}$)
8:         $\mathcal{E}_{\text{new}}$.Add($E^{\text{trial}}_{\text{new}}$)
9:     end if
10: end for
11: $c =$ validstatecounter$/n$
12: return $\{c, \mathcal{E}_{\text{near}}, \mathcal{E}_{\text{new}}\}$
```

– *Propagate:* Returns a new state $E^{\text{new}}_{\text{trial}}$ by applying Eq. (1) to state E_{trial} using ODE as state propagator, which takes care of all the kinodynamic and physics-based constraints.
– *StateValidityChecker:* Checks the validity of the new state $E^{\text{new}}_{\text{trial}}$ as described above.

The set of valid states and confidence index are returned for further processing.

Uncertainty region update The uncertainty region is updated upon the occurrence of robot-object or object-object interactions, as sketched in Algorithm 3. The following functions are used:

– *displacedObjects:* Evaluates for the states in the set $\mathcal{E}_{\text{new}}$ if the objects poses differ from the corresponding in $\mathcal{E}_{\text{near}}$, and returns the number of times this is true.
– *ComputeMeanPose:* Computes for each object i the mean pose $\mathbf{p}^{\text{new}}_i$, from the pose $\mathbf{p}^{E_j}_i$ of each state $E_j \in \mathcal{E}_{\text{new}}$, i.e.:

$$\mathbf{p}^{\text{new}} = (\mathbf{p}^{\text{new}}_1, \dots, \mathbf{p}^{\text{new}}_{j+k}) \quad \text{with} \quad \mathbf{p}^{\text{new}}_i = \frac{1}{|\mathcal{E}_{\text{new}}|} \sum_{\forall E_j \in \mathcal{E}_{\text{new}}} \mathbf{p}^{E_j}_i \tag{2}$$

Algorithm 3 UpdateUncertaintyRegion($\mathcal{E}_{\text{near}}$, $\mathcal{E}_{\text{new}}$, $\boldsymbol{U}_{\text{E}_{\text{near}}}$)

```
1: if displacedObjects($\mathcal{E}_{\text{near}}$, $\mathcal{E}_{\text{new}}$) != NULL then
2:     $\mathbf{p}^{\text{new}} \leftarrow$ ComputeMeanPose($\mathcal{E}_{\text{new}}$)
3:     $\boldsymbol{\nu}^{\text{new}} \leftarrow$ ComputeDeviation($\mathcal{E}_{\text{new}}$, $\mathbf{p}^{\text{new}}$)
4:     return $\mathcal{N}(\mathbf{p}^{\text{new}}, \boldsymbol{\nu}^{\text{new}})$
5: end if
6: return $\boldsymbol{U}_{\text{E}_{\text{near}}}$
```

– *ComputeDeviation:* Computes for each object i the deviation of the poses of the object for the states in $\mathcal{E}_{\text{new}}$, i.e.:

$$\boldsymbol{\nu}^{\text{new}} = (\nu_1^{\text{new}}, \ldots, \nu_{j+k}^{\text{new}}) \qquad \text{with} \qquad \nu_i^{\text{new}} = \sqrt{\frac{1}{|\mathcal{E}_{\text{new}}|} \sum_{\forall E_j \in \mathcal{E}_{\text{new}}} [\mathbf{p}_i^{E_j} - \mathbf{p}_i^{\text{new}}]^2} \quad (3)$$

Note that the uncertainty region is only updated when interactions occur.

4.3 Kinodynamic Motion Planner

For planning the trajectory, an RRT planner is used. It is a sampling-based kinodynamic motion planner that efficiently explores high-dimensional state spaces [20] by growing a tree rooted at a start state. For the growth, a state $\boldsymbol{x}_{\text{rand}}$ is randomly sampled, and the node of the tree that is nearest to $\boldsymbol{x}_{\text{rand}}$ is selected as $\boldsymbol{x}_{\text{near}}$. From that state, randomly sampled controls are applied for a randomly sampled time duration, and the validity-check is run at each step to find collision-free paths. The control that generates a state closest to $\boldsymbol{x}_{\text{rand}}$ is chosen. The newly generated node is called $\boldsymbol{x}_{\text{new}}$ and is added as a vertex, and the control as the edge connecting $\boldsymbol{x}_{\text{near}}$ to $\boldsymbol{x}_{\text{new}}$. The process continues until a state is found within the goal region, or a predefined threshold planning time has elapsed. The present proposal modifies this algorithm by:

a) Introducing a steering method based on the pose of the end-effector, i.e. to steer the tree growth a pose $\boldsymbol{p}_{\text{rand}}$ of the end-effector is randomly sampled and the node of the tree to be expanded is the one that corresponds to a configuration of the robot with the end-effector nearest to $\boldsymbol{p}_{\text{rand}}$. Moreover, a sampling bias is considered towards configurations that place the origin of the gripper reference frame within the target region (a bias index $\mathcal{B} \in [0, 1]$ determines the ratio of the configurations explicitly sampled within the target region).
b) Introducing a knowledge-based state-validity checker to reject states that may lay to a failed execution (Sec. 4.2).
c) Introducing a confidence index to filter out not robust enough controls (Sec. 4.2).
d) Updating the pose uncertainty when interactions occur (Sec. 4.2).

The planning process for the proposed approach is outlined in Algorithm 4. As an input it takes the semantic knowledge stored in the form of ontologies[1] $\mathcal{K}_{\text{S}}$, the initial

[1] https://sir.upc.edu/projects/ontologies/

Algorithm 4 Physics-based Motion Planner for Grasping

```
Input: Semantic knowledge $\mathcal{K}_\text{S}$, Initial state $E_\text{init}$, Initial pose uncertainty region $\boldsymbol{U}_{\text{E}_\text{init}}$, Target object $\mathcal{O}_m^\text{target}$, Time threshold $T_\text{max}$, Bias index $\mathcal{B}$, Robustness threshold $R_\text{threshold}$
Output: A sequence of controls to move the robot to a pre-grasp goal configuration
 1: $\mathcal{K}_\text{m} \leftarrow$ReasoningProcess($\mathcal{K}_\text{S}, \mathcal{O}_m^\text{target}, E_\text{init}$)
 2: $\mathcal{R}_\text{goal} \leftarrow$ComputeEndEffectorGoalRegion($\mathcal{O}_m^\text{target}$)
 3: $\mathcal{T}$.init($E_\text{init}$)
 4: while planning_time $< T_\text{max}$ do
 5:     $\boldsymbol{p}_\text{rand} \leftarrow$SampleRandomPose($\mathcal{K}_\text{m}, \mathcal{B}$)
 6:     $E_\text{near} \leftarrow$SelectNodeToExpand($\mathcal{T}, \boldsymbol{p}_\text{rand}$)
 7:     $\{E_\text{new}\ u_{\Delta t}\} \leftarrow$SearchControls($E_\text{near}, \mathcal{K}_\text{m}$)
 8:     $\{c, \mathcal{E}_\text{near}, \mathcal{E}_\text{new}\} \leftarrow$ConfidenceIndex($\boldsymbol{U}_{\text{E}_\text{near}}, u_{\Delta t}, \mathcal{K}_\text{m}$)
 9:     if $c > R_\text{threshold}$ then
10:         $\mathcal{T}$.AddVertex($E_\text{new}$)
11:         $\mathcal{T}$.AddEdge($E_\text{near}, E_\text{new}, u_{\Delta t}$)
12:         $\boldsymbol{U}_{\text{E}_\text{new}} \leftarrow$ UpdateUncertaintyRegion($\mathcal{E}_\text{near}, \mathcal{E}_\text{new}, \boldsymbol{U}_{\text{E}_\text{near}}$)
13:         if GetEndEffectorPose($E_\text{new}$) $\in \mathcal{R}_\text{goal}$ then
14:             return Path$\leftarrow$RetrievePath($\mathcal{T}$)
15:         end if
16:     end if
17: end while
18: return NULL
```

state of the environment E_init that contains the initial state of the robot and the objects, the initial pose uncertainty region $\boldsymbol{U}_{\text{E}_\text{init}}$, the target object $\mathcal{O}_m^\text{target}$, the time threshold T_max, the bias index $\mathcal{B}$, and the robustness threshold $R_\text{threshold}$. As an output it returns a trajectory from the initial state of the robot $\boldsymbol{x}_\text{init}$ to a goal state $\boldsymbol{x}_\text{goal}$ that places the robot end-effector in the goal region $\mathcal{R}_\text{goal}$. The key parts of the proposed algorithm are:

1. *Initialization* (lines 1-3): It includes the reasoning process to determine the manipulation knowledge, as detailed in Sec. 4.1, and the determination of the region $\mathcal{R}_\text{goal}$ where the end-effector should lie to grasp the object by closing the fingers (i.e. the pose uncertainty of the target object must lie within the gripper aperture, for any configuration within $\mathcal{R}_\text{goal}$).
2. *Tree steering* (lines 5-6): It includes the sampling of a pose of the end-effector $\boldsymbol{p}_\text{rand}$ using a region-biased sampler, and the determination of the node of the tree, called E_near, with a corresponding end-effector pose nearest to $\boldsymbol{p}_\text{rand}$ (a weighted distance measure between translations and rotations is used).
3. *Control selection* (lines 7-8): It first selects a control using a direct control sampling [21], i.e. it samples of a set of random controls and time durations, applies them to E_near using the state propagator and the validity-checker, and selects the one that generates a best valid state E_new (in our case the one that brings the end-effector pose closest to $\boldsymbol{p}_\text{rand}$). Then, the robustness of the chosen control is evaluated with a confidence index as detailed in Sec.4.2.
4. *Tree growing* (lines 9-16): Any control with a confidence index above a given threshold, is added as edge and the generated state as a vertex (lines 10-11). Also the uncertainty region is updated (line 12) as detailed in Sec. 4.2., and if the end-

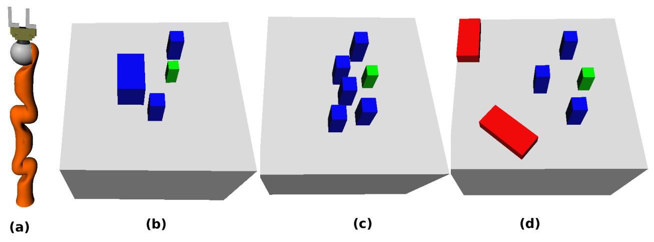

Fig. 4: (a) The 7 degree-of-freedom KUKA robot; (b,c,d) Scenarios where the path to the target object (in green) is blocked by manipulatable objects (in blue) and fixed objects (in red).

effector has reached the goal region then the path is retrieved as a sequence of controls with the corresponding time durations.

5 Implementation, results and discussion

The proposed approach is implemented using *The Kautham Project* [22], an open source motion planning framework that allows to plan under geometric, differential and physics-based constraints, developed in C++ (https://sir.upc.edu/projects/kautham). It uses Open Motion Planning Library (OMPL) [21] for the sampling-based motion planners. The OMPL provides the integration with ODE to handle the physics-based constraints during forward propagation. We implement the variant of kinodynamic RRT that take advantage of high-level reasoning process to guide the planner in an efficient way. The high-level reasoning process is performed over semantic knowledge using Prolog. It is done using the Knowrob software which is a tool to access the OWL knowledge by providing predicates [23].

The proposal is validated using a KUKA lightweight robot with a two-fingers gripper for the three different scenarios presented in Fig. 4. The scenes consist of manipulatable objects (in blue), fixed objects (in red) and a target object (in green). The goal in the presented scenarios is to move the gripper to a pre-grasping configuration to grasp the target object. Since no collision-free trajectory exist from start to goal, to clear the path towards the goal, the robot has to manipulate the objects in a controlled way. The computed plan should be robust enough to be successfully executed in the presence of object pose uncertainty. Fig. 5 depicts the sequence of executions of an example scenario, the computed plan is executed by varying the pose of the object that must be pushed away in order to reach the goal.

The key contribution of this study is the use of high-level knowledge and reasoning process to efficiently plan grasping motions in the presence of object pose uncertainty. To cope with uncertainty, the robustness of the selected controls is evaluated using the confidence index, which is computed by repeatedly applying the control on a set of sampled states and, from the outcomes, estimating the probability of finding valid resultant state. This is a computationally expensive procedure, that is added to the computationally cost of the direct control sampling used in any kinodynamic RRT. The planning

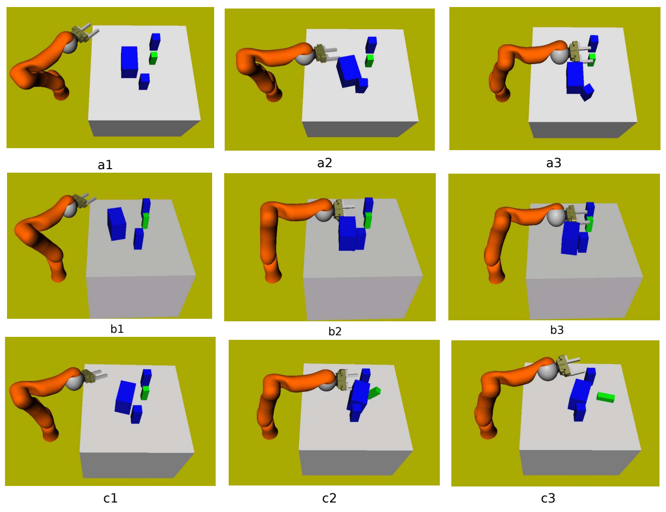

Fig. 5: Snapshots of three task executions with different object poses: the first two rows correspond to successful executions, whereas the last one corresponds to a failed execution. Video:https://sir.upc.edu/projects/kautham/videos/RAL-2.mp4

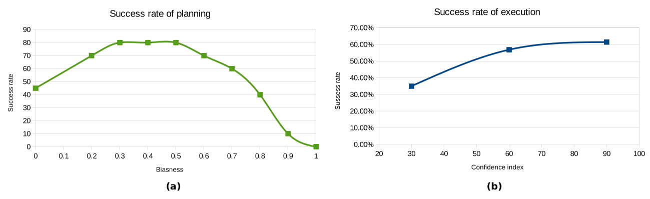

Fig. 6: (a) shows the success rate of the planner using different values of $\mathcal{B}$. (b) shows the success rate of the execution by varying initial poses of the objects for different values of confidence index.

time can be reduced, however, by choosing an adequate region bias or by lowering the robustness threshold. As shown below, in the former case the chosen value can also affect the planning success rate and, in the later case, the chosen value may affect the execution success rate.

The bias index is a critical parameter since its value significantly affects the efficiency of the planner. The planning success rate of the proposed approach is evaluated by varying the bias index between 0 and 1. For each value of $\mathcal{B}$, 20 queries were launched (using the example scenarios), and setting the maximum planning time to 400s. The simulation was run on an Intel Core i7-4500U 1.80GHz CPU with 16 GB memory. The success rate, computed based on the average number of successful runs, is shown in Fig. 6-a. The highest success rate for all the example scenarios is obtained by setting $0.25 \leq \mathcal{B} \leq 0.5$. Running the ontological physics-based motion planning approach [18] the success rate was only around 3%, due to the fact that this planner does

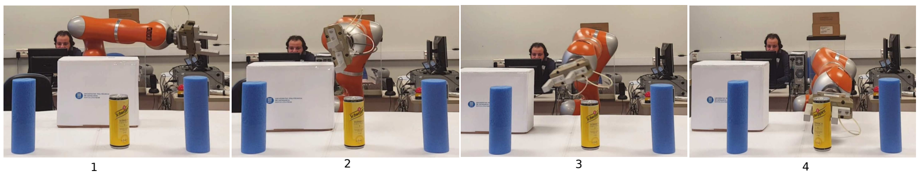

Fig. 7: Execution on real robot for the nominal initial state, goal is to grasp the yellow cylinder. The path to the goal is obstructed with the white box. Video:https://sir.upc.edu/projects/kautham/videos/RAL-1.mp4

not reason on the post-effect of the interactions and since obstacles are very close to the target object, they end by displacing it thus preventing the success of the task execution[2].

Regarding the robustness threshold, motion plans have been computed for values equal to 30%, 60%, and 90%, with $\mathcal{B}$ fixed to 0.3. Then, each computed plan has been executed 20 times in simulation by varying the initial state of the environment. For small deviations in the pose, the computed plans always executed successfully, whereas for the large deviations, the computed plan lead sometimes to unsuccessful results. The success rate of the execution for different values of confidence index is depicted in Fig. 6-b. The execution success rate improves with the robustness threshold, being the effect more relevant when changing from from 30 to 60, and more slightly when increasing up to 90. The computed plan for a scenario similar to the first one has also been executed with the real robot as shown in Fig. 7. The same behavior has been observed.

6 Conclusions

A knowledge-oriented physics-based motion planning approach is proposed for planning the grasping motions in cluttered environments. The semantic knowledge about the scene is stored using OWL ontologies. A knowledge-based reasoning process is proposed that compute the target region and uncertainty regions. The computed regions are used by the sampling-based kinodynamic motion planner that performs the region bias state sampling to plan efficiently, and by the validity checker that considers as non-valid those states that may prevent the success of the task. The proposed approach is validated for different scenarios. The results show a significant improvement in the execution success rate. As a future work, the proposed approach will consider uncertainty in the interaction parameters, like friction coefficients or force directions.

References

1. A. Bicchi and V. Kumar, “Robotic Grasping and Contact: A Review.” in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2000, p. 348353.

[2] Video: https://sir.upc.edu/projects/kautham/videos/RAL-3.mp4

2. R. Suárez, M. Roa, and J. Cornellá, “Grasp quality measures,” Technical University of Catalonia, Tech. Rep., 2006.
3. B. Siciliano and O. Khatib, *Springer handbook of robotics*. Springer Science & Business Media, 2008.
4. L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE Trans. on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.
5. S. M. LaValle, “Rapidly-exploring random trees: A new tool for path planning,” 1998.
6. M. Dogar and S. Srinivasa, “Push-grasping with dexterous hands: Mechanics and a method,” in *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, October 2010.
7. M. Stilman and J. J. Kuffner, “Navigation among movable obstacles: Real-time reasoning in complex environments,” *Int. Journal of Humanoid Robotics*, vol. 2, pp. 479–503, 2005.
8. M. Dogar and S. Srinivasa, “A framework for push-grasping in clutter,” in *Robotics: Science and Systems (RSS)*, July 2011.
9. C. Rodríguez, A. Montaño, and R. Suárez, “Planning manipulation movements of a dual-arm system considering obstacle removing,” *Robotics and Autonomous Systems*, vol. 62, no. 12, pp. 1816–1826, 2014.
10. M. Dogar, K. Hsiao, M. Ciocarlie, and S. Srinivasa, “Physics-based grasp planning through clutter,” in *Robotics: Science and Systems (RSS)*, 2012.
11. N. Kitaev, I. Mordatch, S. Patil, and P. Abbeel, “Physics-based trajectory optimization for grasping in cluttered environments,” in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2015, pp. 3102–3109.
12. C. Erwin, “Bullet physics library, http://bulletphysics.org,” http://bulletphysics.org, 2013.
13. B. Luders, M. Kothari, and J. How, “Chance constrained RRT for probabilistic robustness to environmental uncertainty,” in *AIAA guidance, navigation, and control conference*, 2010, p. 8160.
14. A. Bry and N. Roy, “Rapidly-exploring random belief trees for motion planning under uncertainty,” in *IEEE Int. Conf. on Robotics and Automation*, 2011, pp. 723–730.
15. V. Pilania and K. Gupta, “Localization aware sampling and connection strategies for incremental motion planning under uncertainty,” *Autonomous Robots*, vol. 41, no. 1, pp. 111–132, 2017.
16. N. E. Du Toit and J. W. Burdick, “Robot motion planning in dynamic, uncertain environments,” *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 101–115, 2012.
17. N. A. Melchior and R. Simmons, “Particle RRT for path planning with uncertainty,” in *IEEE Int. Conf. on Robotics and Automation*, 2007, pp. 1617–1624.
18. Muhayyuddin, A. Akbari, and J. Rosell, “Ontological physics-based motion planning for manipulation,” in *IEEE Int. Conf. on Emerging Technologies Factory Automation (ETFA)*, 2015, pp. 1–7.
19. G. Antoniou and F. van Harmelen, “Web Ontology Language: OWL,” in *Handbook on Ontologies in Information Systems*, S. Staab and R. Studer, Eds. Springer-Verlag, 2004, pp. 67–92.
20. S. M. LaValle and J. J. Kuffner, “Randomized kinodynamic planning,” *The Int. Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.
21. I. A. Şucan, M. Moll, and L. E. Kavraki, “The Open Motion Planning Library,” *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012.
22. J. Rosell, A. Pérez, A. Aliakbar, Muhayyuddin, L. Palomo, and N. García, “The kautham project: A teaching and research tool for robot motion planning,” in *IEEE Int. Conf. on Emerging Technologies and Factory Automation*.
23. M. Tenorth and M. Beetz, “Knowrob knowledge processing for autonomous personal robots,” 2009, pp. 4261–4266.