Multiple Object 3D-Mapping using a Physics Simulator

Neeti Wagle, Nikolaus Correll
{neeti.wagle, nikolaus.correll}@colorado.edu

Technical Report CU-CS-1069-10

July 2010

Department of Computer Science
University of Colorado at Boulder
430 UCB
Boulder, CO 80309-0430.

Multiple Object 3D-Mapping using a Physics Simulator

Neeti Wagle and Nikolaus Correll

***Abstract*— This paper presents a novel method for mapping multiple objects such that their positions are consistent with physics laws. We propose the use of a physics engine to help the robot understand the behavior of rigid bodies and their interaction in the real world. We show how pose estimates obtained from other tracking or modeling techniques can be used in conjunction with the physics simulator in order to reject physically impossible hypotheses in a manner similar to human cognition. We present three different examples which exhibit the advantages of such a paradigm for constructing geometrically and physically sound models of the world.**

I. INTRODUCTION

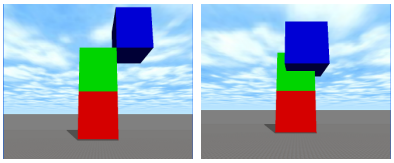
Navigation, exploration and manipulation are the most important high level tasks for a robot. In order to perform these tasks efficiently a robot needs to uniquely identify objects, determine their position in the environment and map them. Several different approaches using laser range finders, monocular vision and stereo vision have been very successful in generating 2D maps and 3D with high fidelity[1].

The majority of the mapping approaches assume only little prior information about the objects, such as coplanarity of polygons and walls being vertically straight[2], [3]. While this is an extremely realistic constraint while mapping unexplored environments, it is too restrictive for non-hostile and interactive environments. For applications like domestic robotics and tour guides we envision an environment where the objects are aware of their own geometry, can communicate with the robot and can send the robot information about their structure. Alternatively, we can use online databases or training models generated by vision algorithms[4], [5], both of which can provide necessary information about the object based on its pose and illumination invariant features. Such a design opens up new avenues which can help address limitations faced in the past.

However, even when the object structure is known and the robot has 6-DOF object poses, 3D mapping can have problems with generating geometrically sound maps. A common and intuitive solution to this problem has been to encode constraints that ensure that maps adhere to laws of physics and geometry[3]. Such approaches are not only tedious but fail to generalize as the complexity of the mapped objects increases.

The task of obtaining physically realistic maps becomes further complicated as the amount of physical contact among objects grows. If each object is being mapped individually, the robot does not have any information about how position estimates of different objects in the scene relate to each other and whether they preserve the overall integrity of the map.

Humans on the other hand can observe a scene and easily reconstruct a valid, harmonious model of multiple objects in the environment via a seamless integration of their conceptual world knowledge with their visual measurements. Fig. 1 shows two configurations of three cubes that cannot exist in the real world. Humans can automatically analyze what they see in these scenes and reject them based on their knowledge of physics. A robot on the other hand has no reason to disregard these configurations. This lack of world knowledge is at the center of the problem. In order to develop a scalable solution the robot would require to emulate a comparable understanding of the nature of physics and the world around it.



(a) Hovering objects (b) Embedded objects

Fig. 1. Invalid object configurations

This paper proposes a novel approach of utilizing a physical simulator as the world knowledge base and analysis engine of the robot. We show how pose hypotheses for different objects can be combined in the physics simulator to recreate a scene which adheres to real world rigid body constraints. By incorporating this powerful tool in a intuitive role, we provide an elegant solution which can be used to replace hardcoded constraints. The block diagram of the system is shown in Fig. 2.

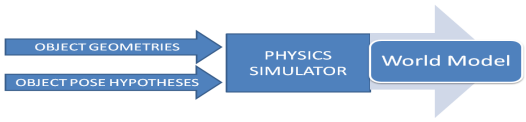


Fig. 2. Block Diagram for Multiple Object 3D Mapping using a Physics Simulator

II. RELATED WORK

Simulators have been used extensively in diverse aspects of robotics. Several simulators like SimRobot[6], USARSim[7] and, Webots [8] provide a cost-effective platform for designing and debugging robot prototypes. Thus, creating a useful simulation involves testing its accuracy and validity by comparing its behavior with that of the real world [9]. In order words, the simulator is used for upstream modeling and validation i.e. replicating real world behavior in computer simulations. In contrast, our approach uses a simulator in a downstream validation process. We make use

of a physics simulator to validate and reject hypotheses that cannot exist in the real world. Thus, we propose the use of a simulator as a pseudo expert system for physics.

Under this paradigm it plays a role which is strongly reminiscent of theory of *qualitative physics* proposed and studied during the 80s and 90s. Qualitative physics aims to study the behavior of mechanisms via simulations and without numerical solutions [10],[11]. The motivaton behind this general and multi-purpose approach ([12]) is to incorporate common sense knowledge of the physical world into the robotic system [11]. Our approach shares this motivation.

However, qualitative physics does not use any mathematical formalism, and looks to explain entire mechanisms through simulations[12]. Our approach avoids both of these pitfalls by employing the physics simulator to only reject impossible behaviors or physical configurations. Using the physical simulation in this limited role helps the system retain control on the solution in a numerical manner while incorporating the aforementioned common sense knowledge.

The method proposed in this paper utilizes the collision detection mechanism in physics simulators to identify invalid multi-object configurations. Rigid body collisions have also been used in grasping to plan the path of a manipulator arm[13]. Their technique introduces rigid bodies for each object's hypotheses in order to refine their task space, register potential space occupancy and prevent collisions between the arm and the object. However, this algorithm focuses on one object at a time and does not consider the physical interaction that may exist between different objects in some configurations e.g. stacks. In contrast, our approach focuses on finding a configuration of all objects in the scene such that no two collide and all poses obey laws of physics.

III. THE PHYSICS SIMULATION PARADIGM

A. Pose Hypotheses and Configurations

While mapping a scene, at any given time, the robot has a 6-DOF pose hypothesis for every object it has sensed or observed. Each of these hypotheses, h can be given by

$$h = (x, y, z, \theta, \phi, \psi)$$

where the first three represent the XYZ co-ordinates of the object and the last three represent its pitch, yaw and roll. The robot may have a single hypothesis or a set of M hypotheses for each object i.e. in an N object scene

$$H_i = \{h_1, h_2, \ldots, h_M\} \quad \forall i \in [1, N], \quad M \geq 1$$

These hypotheses represent the robot's beliefs about the objects in the scene. The degrees to which the robot believes these hypotheses are given by their posterior probabilities p_k. Thus, the robot's beliefs about an object can be completely represented as $H_i = \{(h_1, p_1), (h_2, p_2), \ldots, (h_p, p_M)\}$

Hypotheses may be obtained from a vision pose estimation algorithm or from a dynamic state estimation technique like particle filtering.

In a world or scene with N objects, a combination of N hypotheses is regarded as one configuration. The space of all configurations is given by $\mathcal{C} = \mathcal{H}_1 \times \mathcal{H}_2 \times \ldots \times \mathcal{H}_N$. Thus, a configuration is an instance of a 3D map of the world with each object, O_i, positioned according to one of its pose hypothesis. j_i, and can be represented as

$$C_k = (h_{1j_1}, h_{2j_2}, \ldots, h_{Nj_N}) \tag{1}$$

When the robot's position is known while mapping the estimation of each object is independent of others[14]. Thus, the probability of a combination of hypotheses is the product of their individual probabilities, i.e. the probability of any given configuration C_k is

$$p(C_k) = p(h_{1j_1}, h_{2j_2}, \ldots, h_{Nj_N}) \tag{2}$$

$$= \prod_{i=1}^{N} p(h_{ij_i}), \text{ where } j_i \in [1, M] \quad \forall i \tag{3}$$

Because each object's pose is determined in isolation the errors in each of the estimations are independent. However, this also prevents any information from being shared between the estimation processes. Hence it is possible for a configuration to contain different types of space violations.

Consider the example where two objects located in close proximity to each other are mapped using two independent pose estimation processes. If the magnitude of the sensor noise is high, it is possible for their hypotheses to map the objects overlapping one another. This is the case shown in Fig. 1(b). Here the poses of different objects are in violation of each other.

Allowing object poses that would not be stable under the influence of gravitational and other kinematic forces can result in unstable configurations where objects are hovering (see Fig. 1(a)) or precariously balanced. This can lead to unrealistic and incorrect scene reconstructions. In order to prevent this, we have to inspect the physical interaction between objects. Independent pose estimation provides no opportunity to identify such object interactions and consequently fails to weed out invalid configurations.

B. Rigid Body Properties

We use two key behaviors of rigid bodies to reject unstable or invalid configurations. We describe each of these in detail

1) Volume separation: Volumes of rigid bodies do not overlap. A given unit of volume cannot be occupied by more than one object simultaneously. Therefore, the only contact between two independent bodies has to be external in nature. A violation of this property is a direct identifier of an invalid configuration.
   Most sophisticated physics engines contain a collision detection system that can detect the presence and nature of collisions between geometries in the simulated world. We use the depth of the collision to detect volume overlap between distinct objects. If the overlap depth exceeds a predefined threshold then the configuration is violating the volume separation rule. This threshold is selected in order to discount for jitter in the model.

2) Gravitational force: Due to gravity there are some poses in which an object cannot be at rest. The gravitational forces dictate the valid poses that an object of a given geometry can have. As a result, an object in an invalid pose will succumb to the gravitational force, undergo motion and reposition itself in a stable pose. When time is advanced in the physics simulator, it exerts gravitational pull on all rigid bodies. This causes any unstable or hovering objects to fall to the solid plane. We measure any such motion in terms of the jitter, which is defined as the Euclidean distance between the resultant object pose, h_i', from its original pose, h_i,

$$J_i = \|h_i - h_i'\|_2$$

Now, if $J > T_J$ where T_J is a predefined jitter threshold, then the original configuration is unstable and thus invalid.

These properties are central to the hypotheses rejection process described next.

C. Hypotheses Rejection

Most physics simulators, like ODE and Bullet, provide tools to create common geometries like boxes, cylinders, spheres. Using these geometries we can generate rigid bodies for each object. We remind the reader that the robot can model objects because the it has access to each object's structural information. We can then reconstruct a configuration by positioning the objects according to their hypotheses.

Our approach uses the physics simulator to eliminate invalid configurations that violate atleast one of the two properties from Subsection III-B. These violations indicate that this combination of hypotheses for the different objects cannot co-exist in harmony. However, it is important to note that this does not invalidate any of the individual hypotheses; each of them may be completely valid in another configuration.

Hypothesis rejection is a two step process. In the first step the configuration is tested for volume separation. If no collisions are detected, then the configuration proceeds to the second test. Here it is considered stable if the objects remain at rest in their hypothesized poses when the simulation is advanced in time.

Using this method of hypotheses rejection, the final world configuration can be determined from the space of all possible configurations using an iterative procedure. However, if there are M hypotheses per object then the number of configurations containing N objects is equal to M^N which is exponential in the number of objects. This makes exhaustive analysis of the configuration space intractable. We tackle the problem of exponential number configurations using a branch and bound approach.

Configurations are created by selecting hypotheses for each object in a decreasing order of probability. This means the first configuration $C_1 = (h_{11}, h_{21}, \ldots, h_{N1})$ (Eqn.1), is the highest probability configuration (Eqn.3) and it represents the root of our branch and bound tree.

Algorithm 1 Hypotheses Rejection

Require:

- number of objects N
- number of hypotheses per object M
- threshold probability T
- hypotheses $H_1, \ldots H_N$
- object geometries

Order $H_i \quad \forall i$ in decreasing order
Max-priority queue $V = \{(h_{11}, \ldots, h_{N1})\}$
List of expanded nodes $E = \{\}$
Solution $S = \emptyset$
repeat
  $v = DeleteMax(V)$
  if $v \notin E$ **then**
    if $p(v) \geq T$ **then**
      $Add(V, Children(v))$
      $Add(E, v)$
    end if
    if $VolumeSeparation(v)$ =true **then**
      if $Stable(v)$ =true **then**
        if $Jitter(S) > Jitter(v)$ **then**
          $S = v$
        end if
      end if
    end if
  end if
until V is $\{\}$ and remaining time > 0
return S

In the branch and bound tree, every node has N children, obtained by replacing one of N objects' hypotheses by the next hypothesis. For example, replacing the j^{th} hypothesis, of say object i, with the $j + 1^{th}$ hypothesis results in new child node. Note that since

$$p(h_{ij}) \geq p(h_{i,j+1}) \quad \forall i, j \tag{4}$$

$$p(C_{parent}) \geq p(C_{child}) \tag{5}$$

This means that as we explore configurations down a path of the branch and bound tree the probability decreases. This property can be used for pruning the tree and configuration space by bounding configuration probabilities below a threshold T.

For branching we maintain two lists for visited and expanded nodes respectively. The expanded nodes list prevents repeated exploration of the same configuration in different branches. The visited nodes priority queue ensures that visited nodes are expanded in decreasing order of probability. In this way using branching we examine the higher probability i.e. more promising configurations earlier.

The first valid configuration found is stored as the solution to the problem. It may be replaced by any consequent valid configurations if they exhibit lesser amount of jitter.

The entire process of hypotheses rejection is outlined in Algorithm 1.

In the event that the robot runs out of time, the branch-

ing strategy guarantees that it abandons lower probability configurations. These lower probability configurations are less likely to be the best global solution or provide a massive improvement over configurations found amongst their ancestors. Thus, by going through configurations in this order we maximize the chances of finding the global solution early on. Hence, the algorithm provides the best solution in the given time and is an anytime algorithm.

IV. RESULTS

We have implemented this approach using the Open Dynamics Engine (ODE) which is an open source physics simulator developed by Russell Smith [15]. ODE provides basic geometries like box, sphere, cylinder for common volumes and also contains joints to aid the creation of complex geometries.

In this section we present three examples which represent different geometric and physical challenges to 3D mapping.

A. Peg in a hole

Some of the early work in robotics focused on the problem of inserting a peg into a hole. The task is complicated by the uncertainty in the peg's pose [16]. Such uncertainty can be reduced by eliminating invalid pose hypotheses by using the physics simulator. For example, using collision detection the physics simulator can identify that the peg(blue) poses in Fig. 3 are not valid since they violate the volume separation rule by overlapping the walls of the square hole (white).

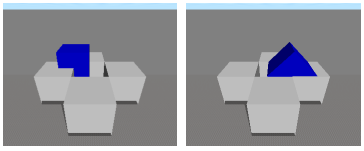


Fig. 3. Invalid peg positions

B. Stacks of Cubes

We present examples of stacked cubes in different configurations where objects have a high level of physical contact. We demonstrate how the iterative procedure (from Subsection III-C) finds a valid final configuration from the particle filter hypotheses for each of the cubes.

The first example, which is a stack of cubes in a vertically aligned form, was constructed from 5 random hypotheses per object. The branch and bound tree for this example is shown in Fig. 4. Each node in the tree is annotated with the code of the configuration C, its probability p, and the physics simulator outcome. A configuration code $C = ijk$ represents a configuration $C = (h_{1i}, h_{2j}, h_{3k})$. Configurations that violated either of the rules are marked as 'Invalid', whereas valid configurations are annotated with their jitter J values.

Fig. 4 shows that the highest probability configuration C_{111} at the root has a collision between the blue and green cubes. Additionally, 2 of the 3 cubes are suspended midair making C_{111} invalid. The nodes are expanded in the non-increasing order of their probabilities. C_{211} is the first visited node that is found to be valid with a jitter value of 0.99; it is recorded as the first solution. However, C_{311} discovered at a later stage exhibits a lower jitter of 0.75 and is selected as the final solution. While most of the invalid nodes in the tree are rejected due to collisions, C_{212} and C_{222} are rejected by the second test because they are unstable, which causes the blue cube to fall to the ground in both cases. Finally, if this tree was extended further all of the leaf nodes except C_{222} would have been expanded; C_{222} has a probability of $0.0001 < T$ and represents a bounded node.

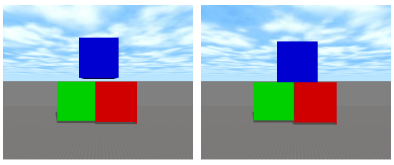


Fig. 5. 3D Mapping for pyramid of three cubes. (a)C=111, Invalid. (b)C=544, J=0.6217

The second example (Fig. 5) increases the complexity of object interactions by introducing an object that is partially balanced on two different objects. In this experiment, we estimated the positions of the centers of three 20cm cubes using a particle filter over 10 iterations(Fig. 6). Traditionally, the best hypothesis of each particle filter would be used to map the cubes at each iteration. Instead, we defined configurations from these particle filter hypotheses and the physics simulator was used to find a final world model. For example, at iteration 7, instead of taking C_{111} to be the 3D map, we ran the hypotheses through the simulator. It showed that not only did C_{111} have a collision between the green and red squares but the blue square was also suspended midair (Fig. 5). On the other hand, C_{544} had no collisions and exhibited a stable and well balanced pyramid of 3 cubes.

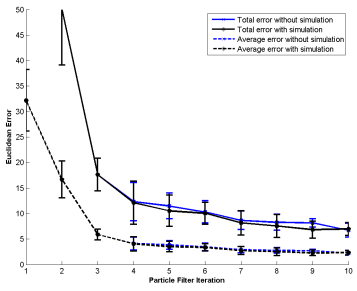


Fig. 6. Noisy 6DOF hypotheses for the 3 20cm cubes were generated by adding noise with $\sigma = 3$ in each dimension. In each run a 1000 particle particle-fiter was used to estimate the cube positions. The above figure shows the total configuration and average object mapping centimeter errors with and without the physics simulator over 20 runs.

We compared the average mapping error per object as well as the total error in the configuration at each iteration, for both C_{111} (blue lines) and the final configurations (black lines) (Fig. 6). While the physics simulator cannot improve the error beyond the quality of the particle filter hypotheses, we can clearly see that a reduction in the mapping error can

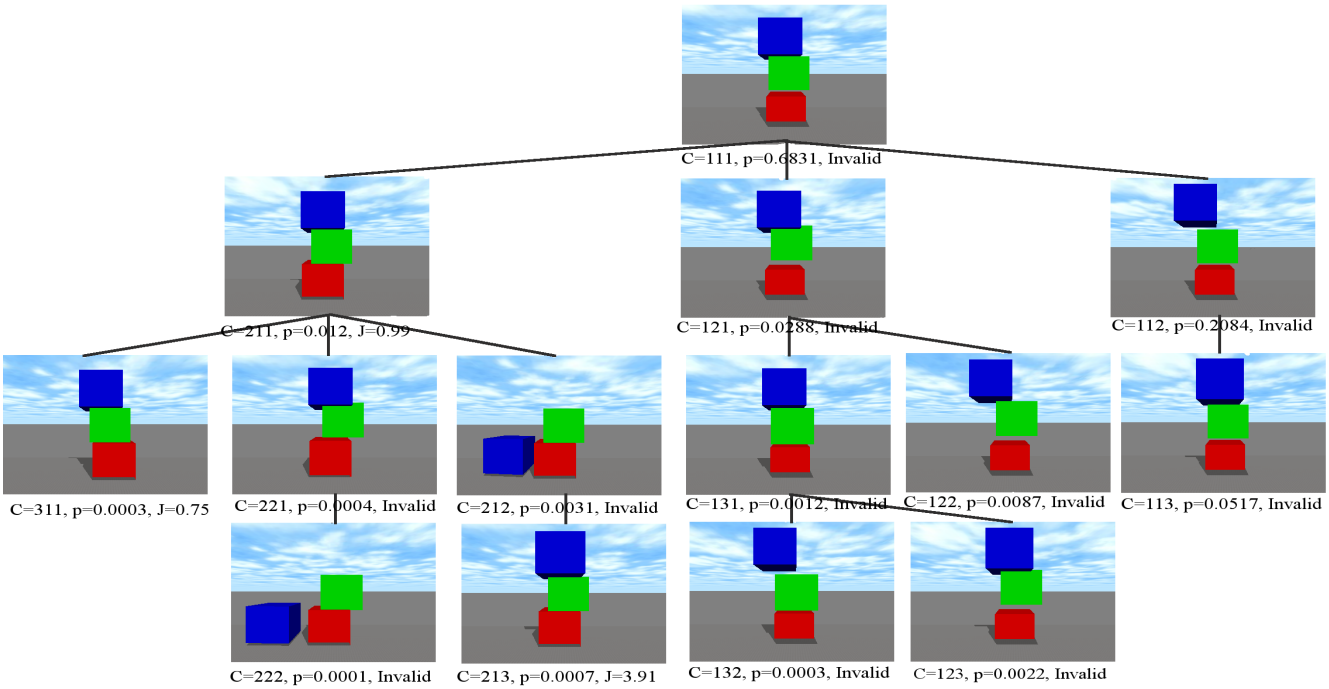


Fig. 4. Branch and Bound Tree. The nodes are expanded in the order 111, 112, 113, 121, 211, 122, 212, 123, 131, 213, 221, 132, 311, 222. Bounding threshold $T = 0.0002$.

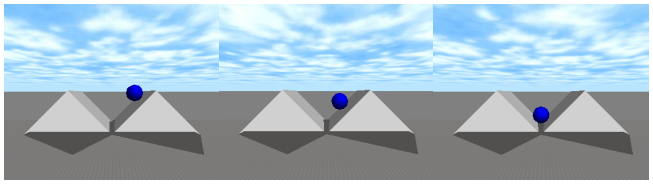


Fig. 7. Sphere mapping using the physics simulator. This figure (l-r) shows the motion of the sphere rolling down the slope starting from the initial position in the leftmost frame until it comes to rest at the minima in the rightmost frame

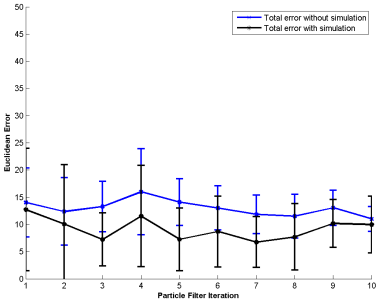


Fig. 8. Error obtained with and without simulation when mapping a 5cm radius sphere with a particle filter over 20 runs. The 6DOF hypotheses were generated with noise $\sigma = 10$ in each dimension.

be obtained by opting for the geometrically and physically hypothesis chosen by the simulator instead of the highest probability hypothesis of each object. Though the savings per object are small for this geometry, the benefits of the reduction in the total congifuration error accumulate as the number of objects in the scene increase. Furthermore, Kruskal-Wallis test on the errors shows results in a p-value of 0.0307. Thus, our method helps obtain lower bounds on the error of the estimation process.

C. Sphere on a slope

In this example, the world consists of a fixed V-shaped groove and a sphere placed at its minima (Fig 7). We consider the case where the pose estimation algorithm has mapped the sphere to a point on the slope instead of at the minima point. A human can look at this configuration and immediately identify that it is impossible. However, the robot with no physics knowledge has no basis to arrive at such a conclusion. If the output of the pose estimation algorithm is now run through the physics simulator for validation it immediately becomes clear that the sphere will roll down the slope to the minima of the groove. Because of this the physics simulator provides a significant improvement over the C_1 hypothesis (Fig. 8). Thus the simulator not only disproves that the sphere can be on the slope but also shows that the valid configuration for these geometries is the sphere resting at the groove's minima.

V. DISCUSSION

Several vision approaches based on SIFT or similar techniques[17], [4], [5] have been proposed for estimating the 6-DOF pose of objects in a scene. Each of these methods use a training phase to generate a model of the objects which is then used in recognition. While these methods localize each object individually and can identify multiple objects

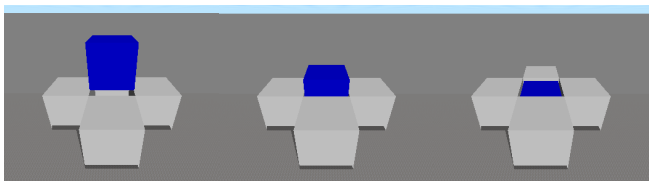


Fig. 9. Fitting peg in the hole using the physics simulator. This figure (l-r) shows the motion of the peg as it is dropped into the hole starting from the initial position in the leftmost frame until it fits in the hole in the rightmost frame

and multiple instances of the same object in the scene, they do not check for the validity of the entire configuration. Our approach can be used to analyze their pose hypotheses and generate physically consistent 3D maps, thus serving as a physical validation of the multiple objects' location hypotheses.

The examples involving cubes show how errors in mapping individual cubes can results in volume overlap problems. Thus, increased proximity poses special challenges for 3D mapping. We have illustrated the ability of the physics simulator approach to find the best overall solution even in challenging configurations where objects are balanced on each other.

The rigid body dynamics change when the object is resting on one of its rounded surfaces. Such surfaces require a completely horizontal plane to come to rest, and roll when placed on a slope. Any configuration which contains a spherical object on a slope is not stable and will exhibit motion and change when time is advanced. Since the original configuration cannot exist, this indicates an error in the position of the spherical object. The third example illustrates how the physics simulator is useful in exposing erroneous pose estimates which cannot exist in the real world due to the action of forces like gravitation.

Furthermore, the physics simulator can play an additional role in problems like the peg-in-the-hole, which involve motion and manipulation. It can provide direct clues about which configurations can lead to the peg fitting into the hole exactly. For example, when the peg from the leftmost frame of Fig. 9 is released, it falls straight into the hole. In this manner results of physical simulation can be used by the robot as positive and negative reinforcements of its current manipulation strategy in a manner akin to human decision making. This example illustrates how the physics simulator is useful in configurations where the nature of physical contact between objects is complex and constrained.

VI. CONCLUSION AND FUTURE WORK

This paper introduces a formal approach to reason about the implications of pose estimates in multi-object settings by relying on physical simulation. As the number of possible hypothesis in a 6-DOF estimation problem grows exponentially with the number of objects, we present a branch-and-bound algorithm that evaluates only the subset of most-likely hypothesis.

This paradigm models object mapping process after the process that humans use for the same task. In doing so, it captures the elegance and scalability of the approach. Another key advantage of this approach lies in transforming a source of complexity into a source of information. Instead of avoiding the interaction between objects it harnesses it to extract realistic limitations of configurations. This results in a mapping solution that is not only based on the estimation process but also consistent internally as well as with real world physical laws.

In future work, we are interested in using an image of the final configuration for visual validation. This would provide feedback via the comparison of this reconstructed world with the camera feed.

REFERENCES

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localisation and mapping (slam): Part i the essential algorithms," *Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[2] S. Thrun, W. Burgard, and D. Fox, "A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping," in *IEEE international conference on robotics and automation*, vol. 1. Citeseer, 2000, pp. 321–328.

[3] D. Hahnel, W. Burgard, and S. Thrun, "Learning compact 3D models of indoor and outdoor environments with a mobile robot," *Robotics and Autonomous Systems*, vol. 44, no. 1, pp. 15–27, 2003.

[4] S. Zickler and M. Veloso, "Detection and localization of multiple objects," *Proc. Humanoids 2006*.

[5] A. Collet, D. Berenson, S. Srinivasa, and D. Ferguson, "Object recognition and full pose registration from a single image for robotic manipulation," in *IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan*, 2009.

[6] T. Laue, K. Spiess, and T. Rofer, "SimRobot-a general physical robot simulator and its application in RoboCup," *Lecture Notes in Computer Science*, vol. 4020, p. 173, 2006.

[7] S. Carpin, M. Lewis, J. Wang, S. Balakirsky, and C. Scrapper, "USARSim: a robot simulator for research and education," in *Proc. of the 2007 IEEE Intl. Conf. on Robotics and Automation (ICRA)*. Citeseer, 2007.

[8] O. Michel, "WebotsTM: Professional mobile robot simulation," *Arxiv preprint cs/0412052*, 2004.

[9] S. Carpin, M. Lewis, J. Wang, S. Balakirsky, and C. Scrapper, "Bridging the gap between simulation and reality in urban search and rescue," *Lecture Notes in Computer Science*, vol. 4434, p. 1, 2007.

[10] J. De Kleer and J. Brown, "A Qualitative Physics Confluences," *Artificial intelligence*, vol. 24, pp. 7–83, 1984.

[11] P. Fishwick and B. Zeigler, "Qualitative physics: towards the automation of systems problemsolving," in *AI, Simulation, and Planning in High Autonomy Systems, 1990., Proceedings.*, 1990, pp. 118–134.

[12] K. Forbus, "Qualitative physics: past present and future," in *Readings in qualitative reasoning about physical systems*. Morgan Kaufmann Publishers Inc., 1989, p. 39.

[13] D. Berenson, S. Srinivasa, and J. Kuffner, "Addressing pose uncertainty in manipulation planning using task space regions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '09)*, October 2009.

[14] K. Murphy, "Bayesian map learning in dynamic environments," *Advances in Neural Information Processing Systems*, vol. 12, pp. 1015–1021, 2000.

[15] R. Smith, "Open dynamics engine," `http://www.ode.org/`.

[16] T. Lozano-Perez, M. Mason, and R. Taylor, "Automatic synthesis of fine-motion strategies for robots," *The International Journal of Robotics Research*, vol. 3, no. 1, p. 3, 1984.

[17] S. Ekvall, F. Hoffmann, and D. Kragic, "Object recognition and pose estimation for robotic manipulation using color cooccurrence histograms," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, vol. 3. Citeseer, 2003, pp. 1284–1289.