Autonomous Realization of Simple Machines

Can Erdogan and Mike Stilman

Institute for Robotics and Intelligent Machines
Georgia Institute of Technology

Abstract. For robots to become integral parts of human daily experience, they need to be able to utilize the objects in their environment to accomplish any range of tasks. In this work, we focus particularly on physically challenging tasks that push the limits on the robot kinodynamic constraints such as joint limits, joint torques and etc. Previously, we demonstrated an autonomous planner that instructs a human collaborator where to place the available objects in the environment to form a simple machine such as a lever-fulcrum assembly. In this work, we report results on the autonomous realization of such a design by the humanoid robot Golem Krang, focusing on the challenges of autonomous perception, manipulation and control.

1 Introduction

The ability to use the available objects in the environment towards accomplishing goals is essential to thriving in challenging circumstances. Everyday examples of tool use include simple machines such as levers and pulleys. The challenge in autonomous design of such simple machines is the space of discrete choices for the component options and the related high-dimensional continuous configuration space of the chosen components.

In previous work [2] [3], we demonstrated the constraint satisfaction approach to assembly design, specifically for robotic manipulation and locomotion. The key idea is to represent the constraints between the components of the design, the robot kinematics and dynamics as generic equality and inequality functions within a nonlinear optimization framework and solve for the global minima, if necessary by random restarts. Such global optimization methods have been used in other fields as well, such as operations research [17] and architecture [19].

In this work, we take the next step towards full autonomy where the humanoid robot, Golem Krang, autonomously manipulates the objects in its environment to construct a simple machine. We present an autonomous planner that perceives the available objects, specifically 15 kg cinder blocks and 10 kg 2-by-4 block blocks (e.g. potential levers), relocates them to the desired configurations output by the constraint planner, and actuates them to flip a 45 kg load. Figure 1 demonstrates key scenes from this scenario such as (a) detection of a cinder block, (b) locomotion with a heavy load, (c) manipulating a lever while subject to multiple constraints and (d) application of force to the lever leading to a successful load motion.

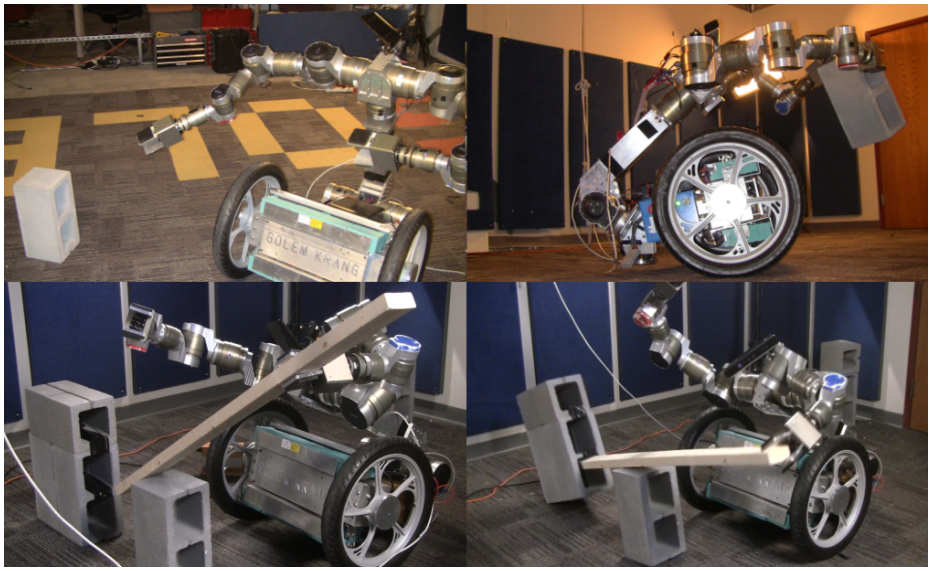

Fig. 1. The mechanical advantage in forces

Significant effort has been demonstrated by [1] [14] [6] to incorporate autonomous agents in human environments. Our work stands out in multiple aspects from the established state of the art. First, Golem Krang is a two-wheeled balancing robot, similar to a segway with two 7-dof robotic arms installed. The challenge with such a platform is the dynamic stability constraint where the robot has to ensure its center of mass is close to the wheel axis at all times as opposed to legged or multi-wheeled platforms. Secondly, to the best of our knowledge, Golem Krang is the tallest and heaviest two-wheeled robot with 150 kg at 1.9 m, a unique property among similar designs [7]. As we expand in Section IV, at this scale, the weight can help with heavy-duty manipulation but complicates the autonomous locomotion of the agent. Lastly, Golem Krang perceives its environment with an onboard RGBD sensor with two degrees of freedom that are manipulated autonomously for gaze control. Autonomous perception and scene recognition has only recently started to gather interest in the humanoid robotics field [13] [10] as opposed to the long established motion capture methods [11].

2 Technical Approach

2.1 Constraint Satisfaction for Simple Machine Designs

The manipulation of multiple objects to achieve a goal can be readily represented in a constraint satisfaction paradigm where the constraints represent the relationships between the design components. In this work, we focus on simple machine designs such as lever-fulcrum assemblies or inclined planes that need to be structurally stable and provide mechanical leverage to their users. Reasoning about such design criteria requires analysis of more detailed concepts such as center of masses, robot kinodynamic constraints and physics principles.

The process for an autonomous planner is composed of three steps. First, from the set of available objects in the environment, it needs to choose a subset that will be incorporated in the structure. Second, the structure components are assigned roles that designate how they should be put together - specifically, the constraints that *bind* one to another. Lastly, the planner needs to configure the objects such that the role constraints and the general design criteria are satisfied.

Component Choices A completeness property for an autonomous planner for structural designs is a crucial advantage for deployment in real-world circumstances (e.g. military or search-and-rescue operations). In emergency situations, when physically challenging that requires creative reasoning about simple machines usually arise, the ability to exhaustively search for all possible solutions and determine if one exists is a critical advantage. Note that we assume every object is used only once in the structure as opposed to the agent changing the structure as its being used.

In choosing a set of design components, an autonomous planner needs to exhaustively search the entire *finite* space of discrete assignments. In comparison to continuous choices, such as object configurations, the discrete nature of the choices (i.e. in or out) makes such a search feasible. Despite the finite space, it is challenging to evaluate every alternative since there is still a combinational number of roles and infinite space of configurations to reason about.

To remedy the computational challenge, pruning strategies and heuristics are significant tools in cutting back the search at the top level. For instance, in construction a lever-fulcrum design, two wooden blocks of the same size (or approximately to a degree of confidence) can be categorized under one class. Similarly, for loads that are known to be significantly heavier than a robot can handle, the longer lever candidates might be priotized in the search process.

Object Roles Imagine a two-step stairs is needed to enable a swarm of rough-terrain vehicles, such as PackBots or RHexs [18, 12], climb a window and survey a building, and some of them have robotic arms that can stack box-like objects. The goal for a planner would be to choose three boxes and stack two of them (e.g. box B on C) such that the swarm can first climb one step and then move on to the two stack, until it reaches the window sill.

In coming up with this solution, two types of choices need to made. First, among the three objects, say A, B, and C, which object would constitute as the first step and which one would be at the top in the second step needs to be decided. Secondly, to place the objects on the ground or on top of another object, a base face needs to be chosen to clarify the object roles and to simplify the representation of the constraints for the object configurations. Note that choosing a base in fact partitions the configuration space of the objects even before the design constraints are considered to prune infeasible assignments.

The goal of the object roles is to specify (1) the relationship of a component with respect to others, and (2) which of its face and edges are used in these connections. Figure 2 [3] demonstrates some outcome assemblies with the triangle prism object acting as fulcrum positioned on different base faces and in contact with the chosen lever on various edges. Similarly, we observe that the lever can be used on at least two of its faces (discounting symmetries) while it is still possible to find a feasible assembly.

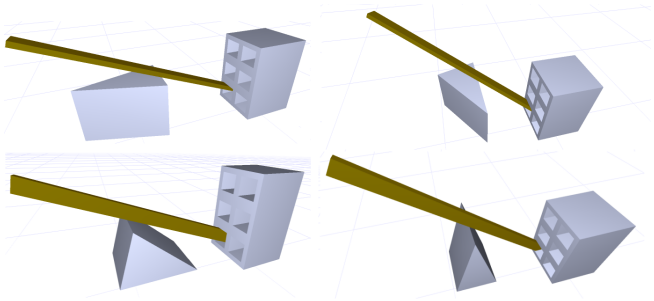

Fig. 2. Different face and edge assignments for the lever-fulcrum assembly. Some assignments, such as using the shortest edge of the lever to connect the fulcrum and the load, may not be feasible due to design constraints and collisions.

The role assignment to available resources has been a thoroughly studied area, starting from classical planning [9, 8], and evolving into operations research [5, 16], and in this work, we adopt the STRIPS representation for the planning process [4]. The idea is to provide the domain knowledge of possible actions to take on the available objects in the environment. For instance, a lever can be *placed* on a fulcrum or a ramp can be *rested* against another object. A minor difference in our framework is that every action induces additional constraints to the assembly configurations and an action can be taken if and only if there exists some configuration that satisfies the accumulated constraints.

Continuous Configuration Space and Constraints The proposed framework accumulates design constraints via a classical planning framework and a feasibility process determines whether the constraints can be satisfied by some

configuration of the components. The idea is that the different type of constraints can all be generically expressed as equality or inequality functions on the space of object and robot configurations. If all the constraints are convex, for instance in a stair or bridge design with simplified robot assumptions, the feasibility can be determined by an efficient simplex algorithm implementation [2]. For the larger set of nonconvex domains, especially when robot kinodynamics are considered, we propose a nonlinear optimization process which minimizes the violation of the constraints and attempts to find an assignment of configurations that satisfies all of them.

A number of design constraints such as stacking a box on top of another one or placing a lever at the edge of a fulcrum can be expressed with geometric projections. Figure 3 demonstrates three types of connections: (1) center of mass-face, (2) edge-face, and (3) face-face. The general idea is that points of interest such as center of mass, endpoints of an edge or vertices of a face are projected to the plane of another face and limits are imposed on its location. For instance, for two objects to be stacked successfully (A), the center of mass of the top object has to lie within the supporting face of the bottom object. Similarly, to ensure an edge is on a face (B), it is sufficient to confine the endpoints of the edge onto the face plane and guarantee there exists a shared point (e.g. cross) between the edge and the face. Lastly, for contact between two faces (C), three points of one of the faces has to lie on the other one and again, a shared point should exist. Observe that geometric contact concepts are easily expressed through equality and inequality expressions on the projections of significant points on the meshes.

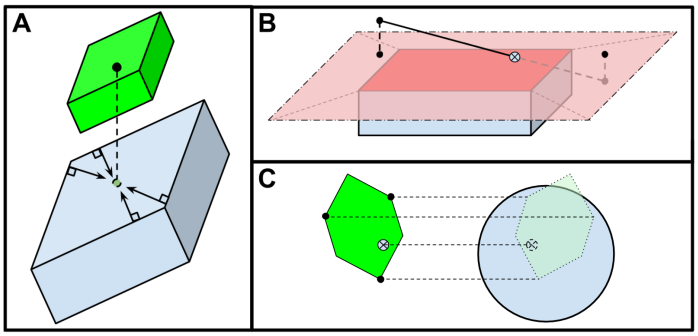

Fig. 3. Visualization of different types of geometric contact constraints

Once the object choices and roles are determined, the equality and the inequality constraints between the assembly components can be gathered into two sets $\mathcal{F}$ and $\mathcal{G}$ respectively. The idea behind using optimization to find feasible samples in the configuration space is based on creating error functions by the violation of the constraints for a sample $\boldsymbol{x}$:

$$f(\boldsymbol{x}) = 0 \;\Rightarrow\; E_f(\boldsymbol{x}) = f^2(\boldsymbol{x})$$
$$g(\boldsymbol{x}) \leq 0 \;\Rightarrow\; E_g(\boldsymbol{x}) = \begin{cases} g^2(\boldsymbol{x}) & \text{if } g(\boldsymbol{x}) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

where $E_f(\boldsymbol{x})$ and $E_g(\boldsymbol{x})$ are the proposed squared error functions. Now, given the constraint sets $\mathcal{F}$ and $\mathcal{G}$, we define the total error:

$$\mathcal{E}(\boldsymbol{x}) = \sum_{f \in \mathcal{F}} E_f(\boldsymbol{x}) + \sum_{g \in \mathcal{G}} E_g(\boldsymbol{x}). \tag{1}$$

Note that $\mathcal{E}(\boldsymbol{x})$ is 0 for some $\boldsymbol{x}$ if and only if configurations $\boldsymbol{x}$ satisfy all the design constraints. Moreover, the global minima is guaranteed be more than or equal to 0 since the function is a sum of squared errors. Then, by using an optimization method, such as Levenberg-Marquardt, the global minima could be found through sampling the space for good initializations.

We outline the overall approach in Algorithm 1 below. Given the available objects, the goal criteria and the available actions, the planner searches in the space of discrete object roles (line 3) and attempts to take actions as long as they lead to feasible configurations (line 9). The forward search accumulates constraints until a feasible design is reached or backtracks.

Algorithm 1: ConstraintPlanner()

Input: *domain*: objects properties and generic actions; *goals*: list of goal literals to be fulfilled; *initialState*: discrete literals and no constraints;
Result: configurations: a feasible value in goal subspace;

```
1  stateStack ←createStack(initialState);
2  while state ← stateStack.pop() do
3      actions ←stateActions(domain);
4      foreach action in the set actions do
5          if action.pres ⊂ state.literals then
6              newConsts ← state.consts ∪ action.consts;
7              for counter = 1 ... MAX_COUNT do
8                  {localMin, confs} ←optimize(newConsts, newSeed(domain));
9                  if abs(localMin) ≤ 1e^-4 then
10                     if goals ⊂ action.afters then return confs;
11                     else
12                         child = {state.literals ∪ action.afters, newConsts}
13                         stateStack.push(child);
14                         break;
15 return ∅;
```

2.2 Humanoid Robot Platform: Golem Krang

Designed and built in the Humanoid Robotics Laboratory, Golem Krang is a 150 kg, 6.2 m segway-like humanoid robot with two wheels that uses a balancing strategy for locomotion and manipulation [15]. Given its unique design, an autonomous planner for functional structures needs to address several hardware constraints. First, the contact point on the lever needs to be reachable by the robot. Second, the robot needs to apply sufficient force to overcome the opposing weight or friction. For Golem Krang, we opt to use only the waist and the wheel motors (Figure 1), and choose to fix the arms with hard mechanical breaks. The motivation is three-fold: (1) the chosen joints have sufficient torque to actuate the simple machine, (2) simpler reasoning about reachable space of the end-effectors, and (3) hardware safety in manipulating hundreds of kgs of objects with an already heavy robot. Lastly, Golem Krang needs to maintain a stable posture before contact is made with the structure.

2.3 Perception

Equipped with a two degree of freedom platform that houses a Microsoft Kinect, Golem Krang can inspect and reason about its environment with visual data. In perceiving the environment, we propose using a light-weight feasure-based recognition approach as opposed to full 3D based approaches that use the entire mesh data such as the iterative closest point algorithm or over-segmentation methods. Figure 4 demonstrates a sample output of the framework.

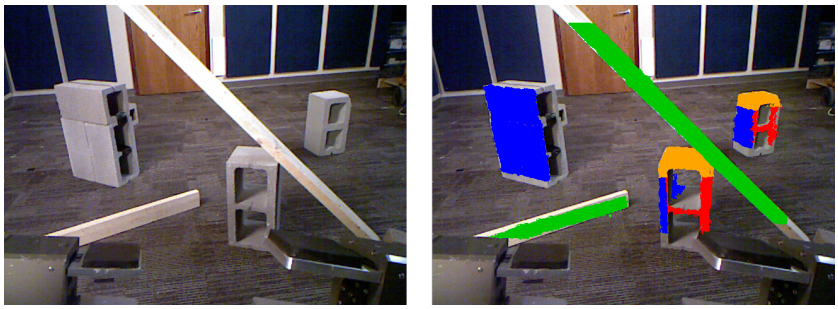

Fig. 4. Cinder block and wooden plates detected as fulcrum and lever objects.

The key assumption of the work is that the planner already knows the mesh of the available objects and with minimal additional feature knowledge, such as the top of a cinder block is at 44 cm from the ground or a lever is at least 2 meters in one dimension, we can speed up the detection. At its current stage, the agent can autonomously detect cinder blocks, the load as shown in Figure 1, the walls and the 2-by-4 levers. To localize objects of interest, we perform a simple scan with the tilt and pan degrees of freedom of the RGBD camera, taking also into account the robot kinematics.

2.4 Locomotion

The primary locomotion strategy for Golem Krang is to balance on its two wheels, keeping its center of mass on the vertical plane through its wheel axis. Modeled as an inverted pendulum, locomotion via balancing has a few advantages over running on both the wheels and the back caster of the robot. First, the footprint of the robot is smaller, 54 cm front to back due to diameter of the wheels while dynamically balancing as opposed to 86 cm when the robot is staticly grounded. Second, the locomotion of the robot is simpler to model since the fixed caster without omnidirectional wheels has different friction properties as the robot spins, moves forward and backward.

The position and posture control is implemented using a proportional derivative controller based on the inertial readings that indicate the robot angle from the vertical and the wheel encoders. In this work, we assume the environment is setup such that the locomotion can be carried out by turning towards the goal position, moving forward and adjusting for the goal orientation - ignoring collisions in the world. To move forward, we use a velocity profile with limits on minimum and maximum acceleration and deceleration.

A significant aspect of the locomotion is the manipulation of heavy objects such as 15 kg cinder blocks and 10 kg wooden plates. To enable stable dynamic balancing, the force-torque sensors at the end-effectors of Golem Krang are used to incorporate the mass of the carried objects into the robot model and update the center of mass position appropriately using forward kinematics. We provide additional insights on heavy object manipulation in the experiments section, specifically on the effect of the object inertia and mass on the system stability.

2.5 Manipulation

References

1. Michael Beetz, Lorenz Mosenlechner, and Moritz Tenorth. Crama cognitive robot abstract machine for everyday manipulation in human environments. In *IROS, 2010*.
2. C. Erdogan and M. Stilman. Planning in constraint space: Automated design of functional structures. *ICRA*, 2013.
3. C. Erdogan and M. Stilman. Incorporating kinodynamic constraints in automated design of simple machines. *IROS*, 2014.
4. Richard E Fikes and Nils J Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3):189–208, 1972.
5. Delbert R Fulkerson. A network flow computation for project cost curves. *Management science*, 7(2):167–178, 1961.
6. Charles C Kemp, Aaron Edsinger, and Eduardo Torres-Jara. Challenges for robot manipulation in human environments. *IEEE Robotics and Automation Magazine*, 14(1):20, 2007.
7. Scott R Kuindersma, Edward Hannigan, Dirk Ruiken, and Roderic A Grupen. Dexterous mobility with the ubot-5 mobile manipulator. In *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pages 1–7. IEEE, 2009.
8. John McCarthy. *Programs with common sense*. Defense Technical Information Center, 1963.
9. Allen Newell and Herbert A Simon. *GPS, a program that simulates human thought*. Defense Technical Information Center, 1961.
10. Koichi Nishiwaki, Tomomichi Sugihara, Satoshi Kagami, Fumio Kanehiro, Masayuki Inaba, and Hirochika Inoue. Design and development of research platform for perception-action integration in humanoid robot: H6. In *Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, volume 3, pages 1559–1564. IEEE, 2000.
11. Vijay Pradeep, Kurt Konolige, and Eric Berger. Calibrating a multi-arm multi-sensor robot: A bundle adjustment approach. In *Experimental Robotics*, 1999.
12. Uluc Saranli, Martin Buehler, and Daniel E Koditschek. Rhex: A simple and highly mobile hexapod robot. *The International Journal of Robotics Research*, 20(7):616–631, 2001.
13. Siddhartha S Srinivasa, Dave Ferguson, Casey J Helfrich, Dmitry Berenson, Alvaro Collet, Rosen Diankov, Garratt Gallagher, Geoffrey Hollinger, James Kuffner, and Michael Vande Weghe. Herb: a home exploring robotic butler. *Autonomous Robots*, 28(1):5–20, 2010.
14. Mike Stilman and James J Kuffner. Navigation among movable obstacles: Real-time reasoning in complex environments. *International Journal of Humanoid Robotics*, 2(04):479–503, 2005.
15. Mike Stilman, Jon Olson, and William Gloss. Golem krang: Dynamically stable humanoid robot for mobile manipulation. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3304–3309. IEEE, 2010.
16. Hamdy A Taha. *Integer programming: theory, applications, and computations*, volume 975. Academic Press New York, 1975.
17. V. Vidal and H. Geffner. Branching and pruning: An optimal temporal pocl planner based on constraint programming. *AI*, 2006.
18. Brian M Yamauchi. Packbot: a versatile platform for military robotics. In *Defense and Security*, pages 228–237. International Society for Optics and Photonics, 2004.
19. Yeung S. Tang C. Terzopoulos D. Chan T. Yu, L. and S. Osher. Make it home: automatic optim. of furniture arrangement. *Siggraph*, 2011.