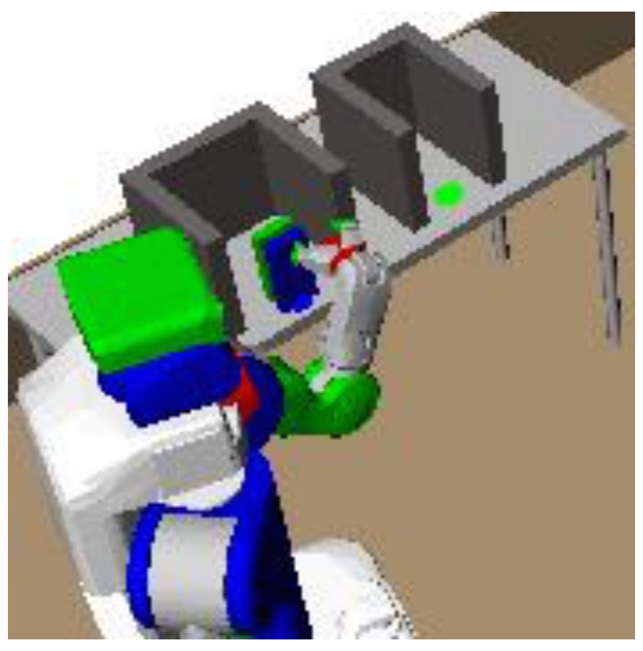
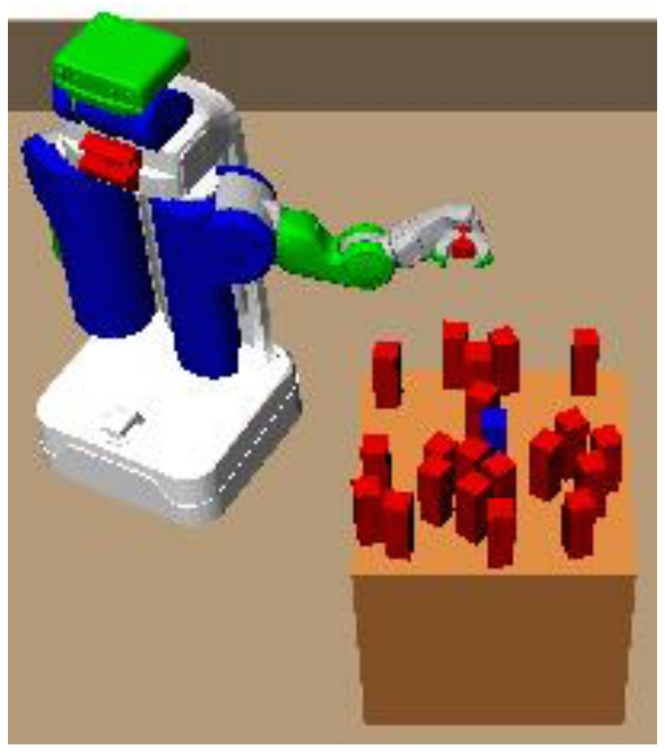
STRIPS Planning in Infinite Domains

Caelan R. Garrett, Tomás Lozano-Pérez, Leslie P. Kaelbling
MIT CSAIL, {caelan,lpk,tlp}@csail.mit.edu

Task and Motion Planning

- Real world planning applications
  - **Continuous variables**
  - **Non-linear dynamics**
- Task and Motion Planning (**TAMP**)
  - Collision, motion, kinematic, and discrete constraints
- STRIPS limited to finite domains
- We **extend STRIPS** to incorporate **external procedures** for modeling these domains
- Also see - "*Sample-Based Methods for Factored Task and Motion Planning*"
  - Transition system formulation
  - **Probabilistic completeness**



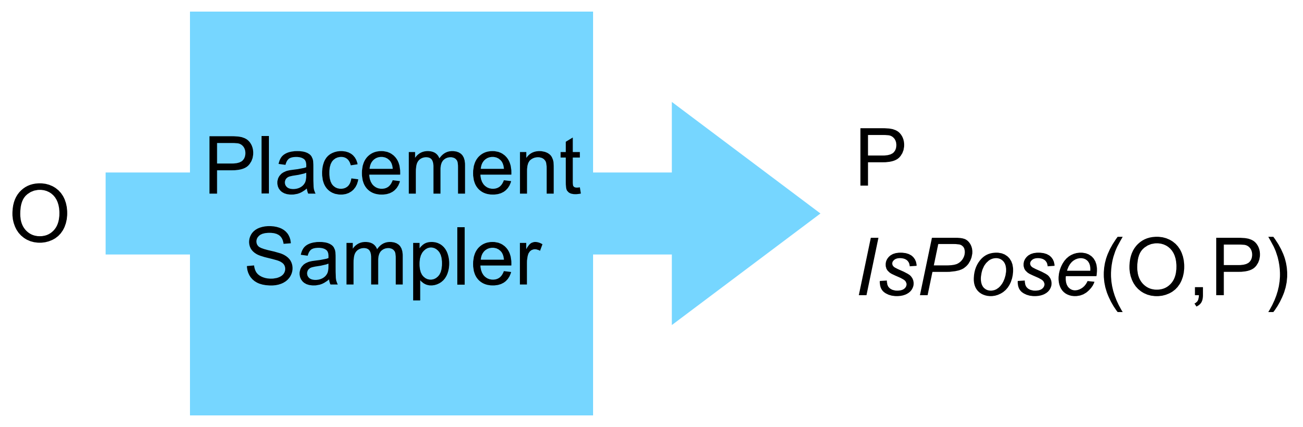


2 Domain-Independent Algorithms

- Reduce to a **sequence of finite domain** problems
  - Automatically **compile** finite domain **to PDDL**
  - Solve using **off-the-shelf planner** (FastDownward)
- **Incremental algorithm**
  - Call finite number of streams and replan
  - Problem - **stream calls often expensive**
- **Focused algorithm**
  - **Plan using streams and actions** to determine which streams could support a solution
    - **Abstract objects** stand in for real objects
  - Call streams along the plan and repeat

STRIPStream

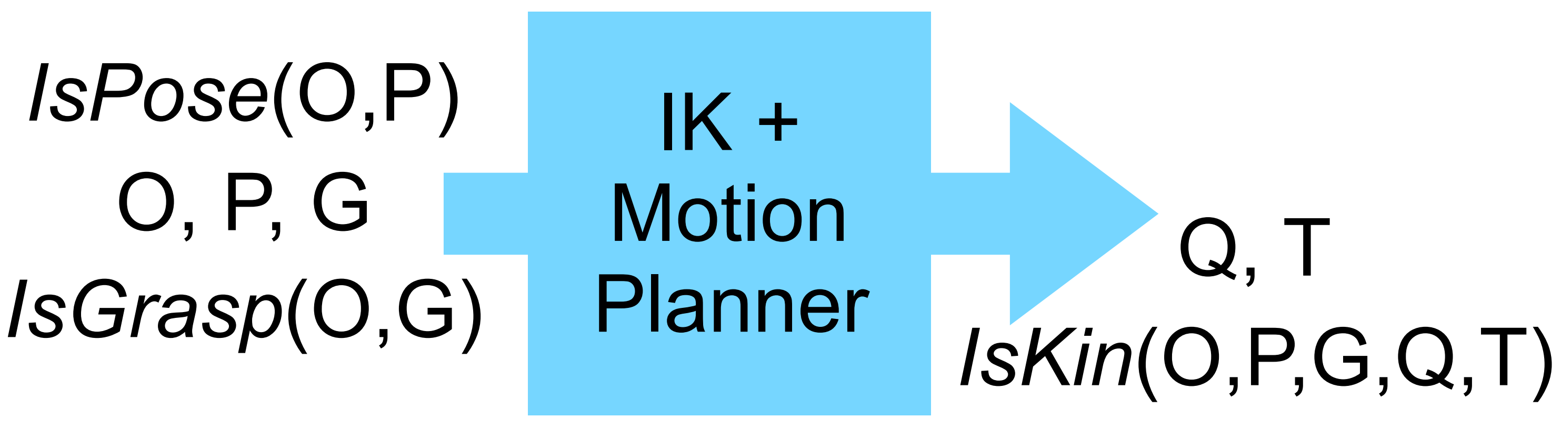
- **STRIPS** (PDDL) **+ Streams**
  - Predicates - fluent and static
  - Actions - params, preconditions, effects
- **Types:** O - BLOCK, P - POSE, G - GRASP, Q - CONF, T - TRAJ
- **Streams** (samplers)
  - Produce **objects** and certify **static predicates**
  - **Conditional generator -** external procedure
- **Conditional Streams -** stream with inputs



O

Placement Sampler

P
IsPose(O,P)

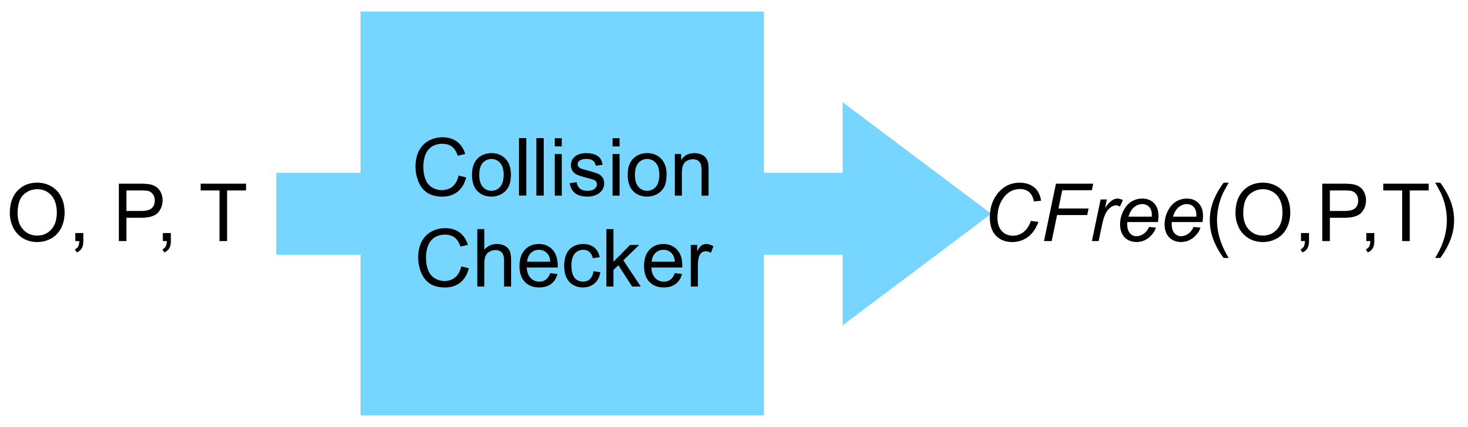


IsPose(O,P)
O, P, G
IsGrasp(O,G)

IK + Motion Planner

Q, T
IsKin(O,P,G,Q,T)

- **Test Streams -** no object outputs



O, P, T

Collision Checker

CFree(O,P,T)

- **Future work** - numerical and temporal planning

Task and Motion Planning in Python

- STRIPStream + Factored Transition System **Software** - https://github.com/caelan/stripstream

Types and Predicates

- Use derived predicates

```
# Types
CONF, TRAJ, REG = Type(), Type(), Type()
BLOCK, POSE, GRASP = Type(), Type(), Type()

# Fluent predicates
AtConfig = Pred(CONF)
HandEmpty = Pred()
AtPose = Pred(BLOCK, POSE)
Holding = Pred(BLOCK, GRASP)

# Static predicates
IsPose = Pred(BLOCK, POSE)
IsGrasp = Pred(BLOCK, GRASP)
IsKin = Pred(BLOCK, POSE, GRASP, CONF, TRAJ)
IsCollisionFree = Pred(BLOCK, POSE, TRAJ)
IsContained = Pred(REG, BLOCK, POSE)

# Derived predicates
Safe = Pred(BLOCK, TRAJ)
InRegion = Pred(BLOCK, REG)

# Parameters
O, P, G = Param(BLOCK), Param(POSE), Param(GRASP)
Q, Q2, T = Param(CONF), Param(CONF), Param(TRAJ)
OB, R = Param(BLOCK), Param(REG)
```

Actions and Axioms

- Safe axiom used to **factor collision checking**

```
actions = [
  Action(name='pick', parameters=[O, P, G, Q, T],
    condition=And(AtPose(O, P), HandEmpty(),
      IsKin(O, P, G, Q, T), AtConfig(Q),
      ForAll([OB], Or(Equal(O, OB), Safe(OB, T)))),
    effect=And(Holding(O, G),
      Not(HandEmpty()), Not(AtPose(O, P)))),
  Action(name='place', parameters=[O, P, G, Q, T],
    condition=And(Holding(O, G),
      IsKin(O, P, G, Q, T), AtConfig(Q),
      ForAll([OB], Or(Equal(O, OB), Safe(OB, T)))),
    effect=And(AtPose(O, P), HandEmpty(),
      Not(Holding(O, G)))),
  Action(name='move', parameters=[Q, Q2],
    condition=AtConfig(Q),
    effect=And(AtConfig(Q2),
      Not(AtConfig(Q))))]

axioms = [
  Axiom(effect=InRegion(O, R), condition=Exists([P],
    And(AtPose(O, P), IsContained(R, O, P)))),
  Axiom(effect=Safe(O, T), condition=Exists([P],
    And(AtPose(O, P), IsCollisionFree(O, P, T))))]
```

Stream Specification

- Generator functions - *sample_poses, sample_grasps, sample_region, sample_motion*
- Boolean functions - *collision_free*

```
cond_streams = [
  GenStream(inputs=[O], outputs=[P],
    conditions=[],
    effects=[IsPose(O, P)],
    generator=sample_poses),
  GenStream(inputs=[O], outputs=[G],
    conditions=[],
    effects=[IsGrasp(O, G)],
    generator=sample_grasps),
  GenStream(inputs=[O, R], outputs=[P],
    conditions=[],
    effects=[IsPose(O, P), IsContained(R, O, P)],
    generator=sample_region),
  GenStream(inputs=[O, P, G], outputs=[Q, T],
    conditions=[IsPose(O, P), IsGrasp(O, G)],
    effects=[IsKin(O, P, G, Q, T)],
    generator=sample_motion),
  TestStream(inputs=[O, P, T],
    conditions=[IsPose(O, P)],
    effects=[IsCollisionFree(O, P, T)],
    test=collision_free)]
```