Variable Level-Of-Detail Motion Planning in Environments with Poorly Predictable Bodies

Stefan Zickler and Manuela Veloso
Computer Science Department
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA

ECAI 2010, August 18th, Lisbon

PEL, Fall 2010

Planning – “thinking/modeling the future”

- Given:
  - State space **X**
  - Initial state $\mathbf{x}_{init} \in \mathbf{X}$
  - Set of goal states $\mathbf{X}_{goal} \subset \mathbf{X}$
  - Set of actions **A**
  - A state transition **MODEL: resulting state x’** of **applying an action a ∈ A to a state x** *(a “model” because, at planning time, the action is not actually applied in the state and its effects are not actually observed)*
- Planning is the search for a sequence of actions $\mathbf{a}_1,\ldots,\mathbf{a}_n$ leading from $\mathbf{x}_{init}$ to a goal state $\mathbf{x}_{goal} \in \mathbf{X}_{goal}$

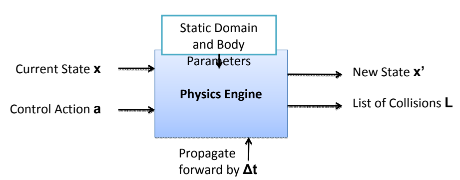
Physics-Based Planning

- Motion planning using domain models that reflect Newtonian physics as encountered in the real world

Stefan Zickler, PhD thesis, May 2010, CMU
www.szickler.net

Physics-Based State Transitions

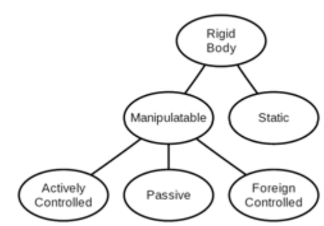
- How do we apply an action and simulate from one state to the next? Use a **Physics Engine**!



Physics-Based Model

- How do we model the physical world?
  - Rigid body dynamics
  - An approximation of real-world physics

Body Types



States

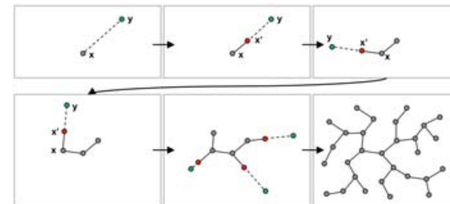
- The system's state **x** contains the state of all rigid bodies at a given point in time **t**:
  x= ⟨**t**, $\mathbf{r}_1$, … , $\mathbf{r}_n$⟩
- A single rigid body's state: $\mathbf{r}_i$= ⟨**p**,**q**,**v**,**w**⟩
  - **p**: position (3D vector)
  - **q**: orientation (3x3 rotation matrix)
  - **v**: linear velocity (3D vector)
  - **w**: angular velocity (3D vector)
- Other body parameters:
  - Shape, mass, material constants, …
  - Constant, thus not considered part of state-space

Actions

- An action **a** is a vector of sub-actions:
  a = ⟨ $\hat{a}_1$, …, $\hat{a}_n$ ⟩
- Each sub-action $\hat{a}_i$ is a ⟨ force, torque ⟩ pair applicable to a corresponding rigid body $\mathbf{r}_i$

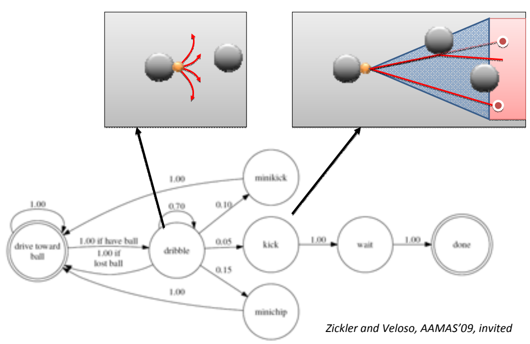
Planning Algorithm: Rapidly-Exploring Random Trees (RRT)

- Create a random sample **y** from some subspace **Y** of **X**
- Find the nearest neighbor **x** in tree **T** using some distance metric
- Expand from **x** toward **y**, creating a new child node y'



[S. LaValle, 1998] [J. Kuffner, S. LaValle, 2000]

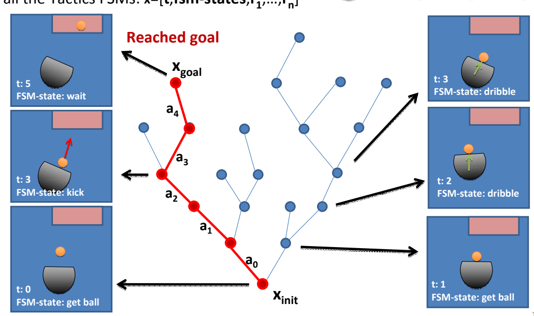
Physics-Based Planning with Non-Deterministic Skills and Tactics



Zickler and Veloso, AAMAS'09, invited

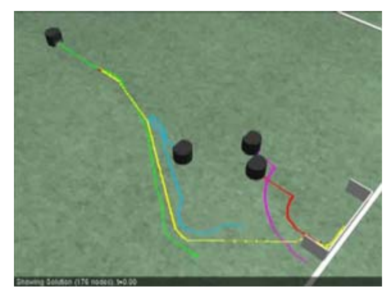
Physics-Based Planning Algorithm

Each node in the tree represents the complete physical state of the world and of all the Tactics FSMs: x=[**t**,**fsm-states**,$\mathbf{r}_1$,…,$\mathbf{r}_n$]



Physics-Based Planning

(still in simulation)



Stefan Zickler – PhD thesis 2010

Planning in a Dynamic World

- At each (re)planning iteration, the planner searches for a complete solution, trying to find a precise trajectory around moving objects , **all the way to the goal state.**
- In domains with high uncertainty, predictions for poorly modeled objects are likely incorrect.
- Generated plans fail early, and most of the plan will **never be successfully executed.**
- A waste of computational planning resources!

Key Question

- How can we make motion planning in environments with poorly predictable bodies more efficient?

Approach

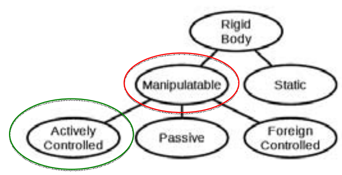
- ***Variable Level-of-Detail (VLOD)*** planning
- The planner should only worry about moving bodies:
  - preventing collisions in the *near future*
  - ignoring collisions in the *far future*

Variable Level-of-Detail (VLOD) Planning

- A method for relaxing the replanning problem by ignoring **details** that occur in the **future**
- **Future**: defined by time horizon threshold: $\mathbf{t_{LOD}}$
  - If $\mathbf{x.t > t_{LOD}}$ then ignore **details** during planning
  - Otherwise, plan with **full detail**

Variable Level-of-Detail Planning

- What is a **detail**?
  - Locally solvable multi-body interactions that do not affect the global topology of the plan if temporarily ignored
  - In our model: Interactions between the **actively controlled** and other **manipulatable** bodies



RRT-Based Planning Algorithm

```
T.AddVertex(x_init); //start with a tree that only contains our initial state
repeat until we give up
    y := SampleRandomState(Y); //generate a random sample from the sampling space Y
    x := NearestNeighbor(T,y); //find nearest neighbor to y in T
    a := Controller(x,y); //generate an action from x toward y
    SetupCollisionMatrix(x); //sets up the currently applicable level of detail
    [x', L] := SimulatePhysics(x, a); //apply and simulate the control action
    if isValidState(x', L) then //check if any constraints have been violated
        T.AddVertex(x'); //add x' as a child of x
        T.AddEdge(x, x', a); //saving the control action a as edge
        if x' is in X_goal then return x'; //have we reached the goal?
    end if
```

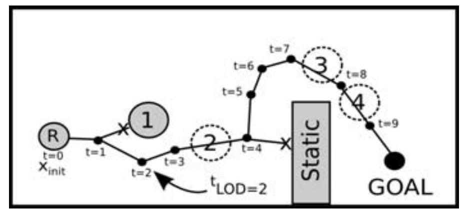
Reconfiguring the Collision Matrix **M**

```
Algorithm 5: SetupCollisionMatrix
// Let m_ij denote the element at the i-th
   row and j-th column of M.
for i ← 1 to n do
    for j ← i to n do
        if i=j then
            m_ij ← 0;
        else
            if (x.t > t_LOD and
            ((r_i ∈ B_Controlled and r_j ∈ B_Manip) or
            (r_j ∈ B_Controlled and r_i ∈ B_Manip))) then
                m_ij ← 0;
                m_ji ← 0;
            else
                m_ij ← 1;
                m_ji ← 1;
```

M is symmetric ($m_{ij}=m_{ji}$)
$m_{ij}=1$: collision between r_i and r_j should be simulated and reported in the list of collisions **L**
$m_{ij}=0$: collision between r_i and r_j should be ignored and not reported in **L**

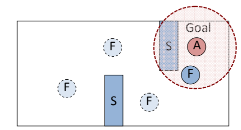
VLOD Example

- **t_{LOD}=2**



R: Controlled Robot Body
1-4: Foreign-Controlled or other Manipulatible Bodies

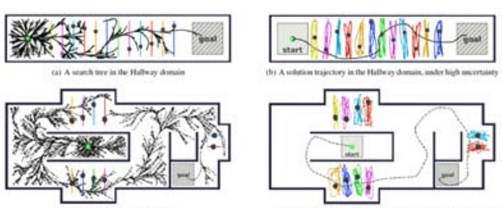
VLOD Example



A: Actively controlled body
F: Foreign controlled body
S: Static body

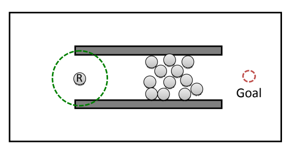
Experimental Domains

- Navigate through a field of rapidly moving foreign-controlled bodies
- Foreign-controlled bodies have simulated uncertainty
- Controllable replanning interval t_{replan}
- Controllable VLOD time horizon t_{LOD}



Assumptions / Limitations

- VLOD assumes that details are **locally solvable** (within t_{LOD}).
- Imagine the following scenario:



- Solution: increase t_{LOD} or redefine what constitutes a detail, depending on the particular domain

Conclusion

- VLOD planning can significantly increase planning speed without increasing number of collisions
- VLOD planning is applicable to many dynamic navigation domains