# Constraint-Aware Navigation in Dynamic Environments

Mubbasir Kapadia, Kai Ninomiya, Alexander Shoulson, Francisco Garcia, Norman I. Badler
SIG Center for Computer Graphics
(a submission to the November 2013 Motion in Games conference)

Our research is in the areas of behavior simulation, robot behavior, path planning, and human-robot interaction.

Our primary goal is to translate a semi-natural formal language behavior specification into a meaningful series of actions that a robot or simulated character can follow. To do this, we express movement behaviors using a structured definition. The command follows the structure at right.

Some of the interesting problems presented are:

- Finding paths that "make sense" given a language representation
- Solving the pathfinding problem quickly enough for realtime interaction, while also being able to deal with a dynamic or constantly-updating world model

Behavior Specification

**Move** to *GoalPosition* [Set of *ConstraintSpecs*].

Behavior Constraint Specification

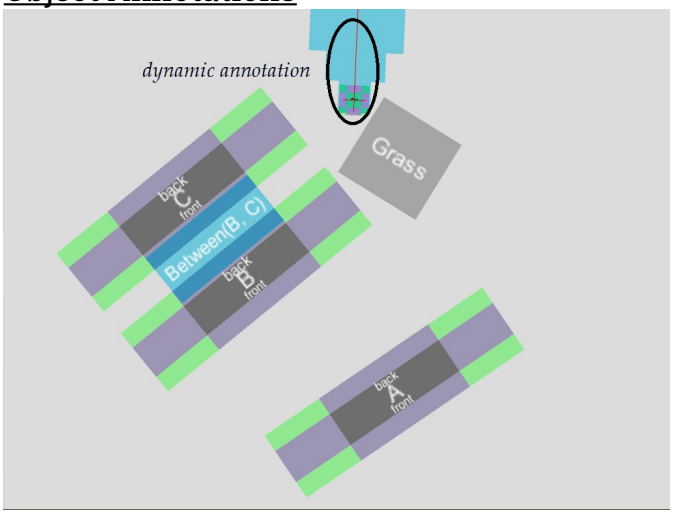
- *Preposition* (either
  - **In**, an area with a hard edge, or
  - **Near**, an area with a soft edge)
- *Object* or *Annotation Object*
  - (an object indicating the area this constraint concerns)
- *Weight w*
  - (a number representing the positive or negative weight of an area)

Obstacles and agents in the world are annotated with additional objects to allow for additional detail in behavior specifications.

Examples of such annotations are shown in the diagram at right:

- LeftOf
- RightOf
- FrontOf
- BackOf
- LineOfSightOf
- Between

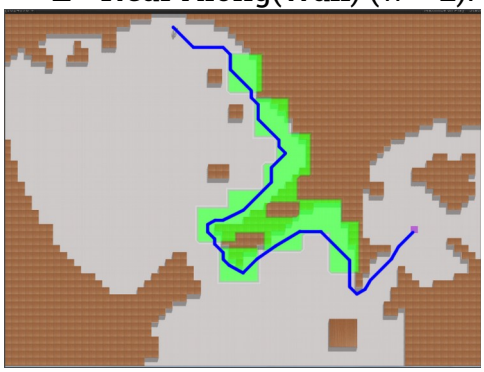
Object Annotations




All of these features allow us to define robust behavior specifications in a computer-understandable form. Examples of behavior specifications are shown on the next page.

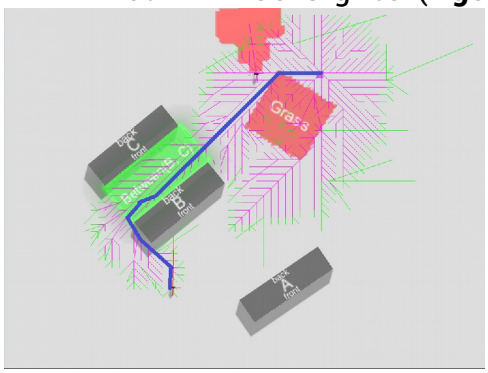
Since constraint specifications are understandable as quantified adjustments to the world space, we can now use path planning to find an optimal path through the space.

- **Move** to **GoalPosition**,
  - **Near** Along(**Wall**) (*w* = 2).

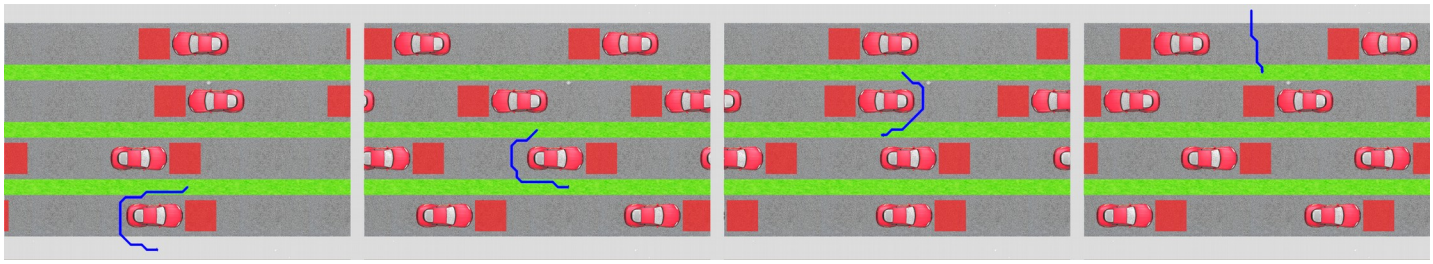


Behavior Constraint Specification Examples

- **Move** to **GoalPosition**,
  - **In** Between(**BldgA**, **BldgB**) (*w* = 1),
  - Not **In** **Grass** (*w* = −1),
  - Not **In** LineOfSightOf(**Agent**) (*w* = -2).




- **Move** to **GoalPosition**,
  - Not **In** FrontOf(**Cars**) (*w* = −5).



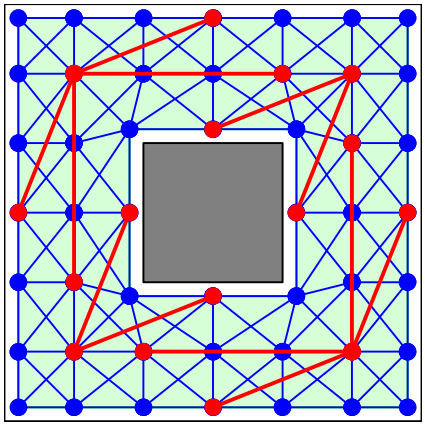
We use Anytime Dynamic A* (AD*), a path planner with the following properties:

- Anytime: the planner finds a suboptimal plan even before it has time to finish planning.
- Dynamic: the planner can quickly repair its plan in the case of environment changes.

This allows our system to work rapidly even in changing environment representations.

An additional improvement to the performance of our system comes from our environment representation. It is represented as a grid-like graph with extra "highway" edges added in, allowing the planner to find better suboptimal paths "anytime."

At right, a small example is shown: blue edges represent the grid portion of the graph while red edges represent the highways.



Future work includes:

- Integrating the project with our new planner framework (currently in progress).
- Using spatial optimizations to speed up operations used in pathfinding.
- Extending the solution to other behaviors (the search is not limited to spacial problems).
- Writing a lazily-evaluated domain so that the graph does not have to be precomputed.