Real-time Artificially Intelligent Ski Racer (RAISR)

To be completed by 2 team members.

Brief Description

We would like to implement an artificially intelligent agent that chooses the best path through a ski course. This problem is non-trivial, because simply choosing the shortest and fastest path down the hill can force the racer into ‘slamming on the brakes’ on a tight turn later on, or even being unable make the next turn. Solving the problem requires representing the ski race as a sequence of decisions, and physically modeling the outcomes. Doing so will frame this as a heuristic search problem, to which techniques studied in CSC384 can be applied. We will adapt the code from the first assignment, in which we were given a framework to solve a heuristic search problem. A goal is for this solver to work in real-time, i.e. performing the heuristic search in the time available between turns, roughly a few tenths of a second.

Brief Evaluation Plan

The metric by which solutions can be evaluated comes directly from the reality of ski racing. Each racer is trying to make it through the course in the shortest amount of time. Agents will be evaluated based on how fast the solutions they produce are.
They will also be evaluated on the running time of the solutions. An agent that can solve this with less computation and in closer to real time will be deemed better, as well. We will define a few test courses of varying difficulty and length that could conceivably be set in a real ski race.
We intend to compare Weighted A* search, and Iterative Deepening A* search, each with a set of possible heuristics.

Team Member Roles

Member 1 will work on problem encoding: fitting the solver.py starter code from assignment 1 to our purposes, basic physical modeling to get state update rules resulting from actions, and modeling the problem as a search problem - translating from a sequence of gates to a discrete search problem that can be solved with heuristic search.

Member 2 will work on solving the search problem: implementing the solvers - IDA* as well as the A1 solvers, and implementing effective heuristics for the solvers and testing their efficacy

This is a rough sketch and as the project goes on, more interplay between the two parts will require the two roles to blend into one another somewhat. However this is still a useful starting point.

A More Detailed Outline

A Brief Explanation of Ski Racing

In a ski race, each racer goes one at a time down a course as fast as possible. The racer with the fastest time wins. A course is a sequence of gates that the skier must navigate around on their way down, in alternating left and right turns.

Here's a video to clarify.

Physical Model

We will implement a simplified model of the complicated physics involved in ski racing that makes it more amenable to optimization with AI. The contours of the mountain will be ignored, and the hill will be a simple inclined plane. The course will be a series of (x, y) coordinates on this plane, which the skier must turn around in alternating fashion.

Three forces will be modeled: gravity (constant downwards force), friction (constant force slowing the skier down), and air resistance (proportional to velocity). These will be computed to determine time between turns, which will be added together to produce the total time for a run.

State Space

The race time will be discretized, with each time step in the simulation being the beginning of a turn. The state of the skier is given to the agent at each time step as:

- Speed
- Total elapsed time in the race
- Position of the remaining gates in the course (relative to the racer)

Actions

We will represent actions as three parameters that describe the upcoming turn: length of the horizontal axis of the turn (L_x), length of the vertical axis of the turn (L_y), and the coordinate along the turn at which the next turn is to be initiated (t). This is depicted in the diagram below (Figure 1):

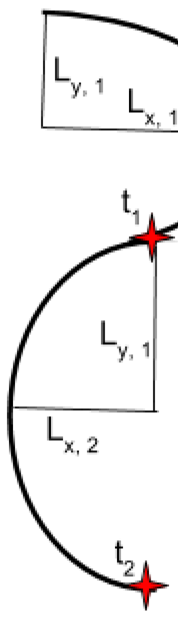

Figure 1: Action Model, showing parameters L_x, L_y and t

This step is challenging and marks a departure from the problems we studied in class. In reality, the action space is continuous, but this cannot be represented simply with the tree structure we studied. The action space must therefore be discretized. To do this, a number of evenly spaced feasible parameter values will be explored. The number of points will based on computational constraints - this will be the branching factor of our search tree.

Initial State

The ski racer starts their search at the top of the slope, with no velocity,

Goal State

The final gate of the course is a special case: rather than choosing the point to start the next turn, this is fixed as the point along the chosen ellipse that intersects with the finish line. At this point **the total elapsed time is the metric that the agent tries to minimize**, analogous to the number of moves made in the Sokoban assignment.

Costs

The action chosen affects the racer's state in multiple ways. Different turns result in different velocities into the next turn, and can take different amounts of time. This problem is non-trivial, because simply choosing the shortest and fastest path down the hill can force the racer into 'slamming on the brakes' on a tight turn later on, or even being unable make the next turn. To clarify, the racer could overshoot the upcoming gate, and have to climb the hill which results in disqualification.

Heuristics

Developing effective heuristics is going to be a significant part of the implementation for the project. However at this early stage we can still suggest some potential ideas that we may use as heuristic functions:

- Current elapsed time
- Estimated distance remaining / velocity, estimating distance using:
    - Vertical straight-line distance to the finish line
    - Sum of euclidean distances between all gates

Potential Extensions

- More realistic skiing physics
    - Pushing out of the start
    - Variation of skier physical parameters, such as weight
    - Different body positions, such as tucking
    - Breaking down the turn into more phases
    - Modeling the contours of the hill beyond a simple inclined plane
- Multiple agents (ski cross!)
- Using an AI search framework that supports continuous action spaces, rather than forcing the problem to be discrete