Creating a Soar agent to play DESCENT

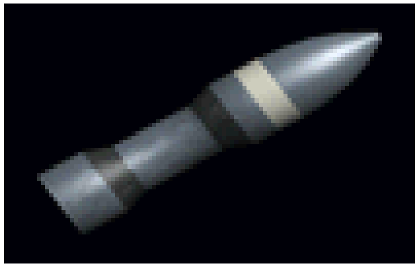
Mike van Lent

Soar Workshop XVII

June, 1997

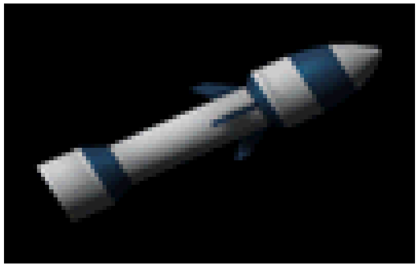
Background

- What is Descent
  - First Person Perspective, Full 3D Arcade Game
  - Doom, Quake, Duke Nukem 3D
    - More complex
- How did this project originate
  - Created by Parallax Inc. (now Outrage)
    - Main office in Ann Arbor
  - Co-founder of Parallax presented a lecture for the games course
  - Impressed by TacAir-Soar
  - Offered to put some effort into interfacing Soar with Descent III



Goals of the Project

- For us
  - Push the Soar architecture in new directions
    - Efficiency, new tasks and new styles of IO
  - Provide a new research domain
  - Soar as a commercial tool for computer game AI?
- For Outrage
  - Learn more about state of the art AI techniques
  - Ideas to improve the AI in Descent III
  - Include Soar in Descent III in some form?
    - Automatic Deathmatch opponent
    - Allow users to program or train new opponents

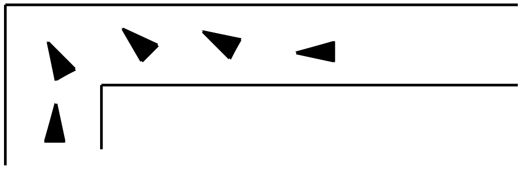


Complexity of Descent

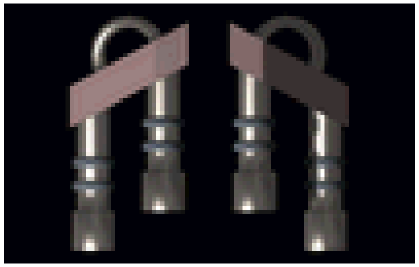
- Full 3D Environment
  - No Gravity, No pre-defined up or down
  - Dynamic lighting
  - Can’t hard code map features
- 23 Controls
  - Movement with 6 degrees of freedom
- 40+ Enemies
  - AI is already impressive (Teamwork, Dodging, Varied Strategies)
- 30+ Weapons and Powerups
- 60+ Tactics on one website guide

Examples Tactics

- Circling
  - Use slide left & turn right (for example) to change position without changing where you’re pointing (aiming)
- Corner Spin
  - Turn a corner by sliding and turning to face backwards

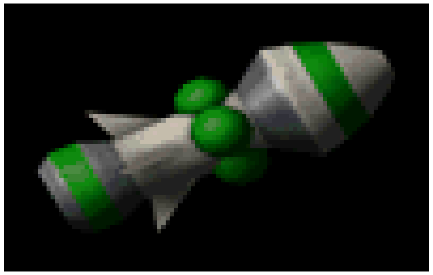


- Hide and Search
  - Hide in a secret room and use a remote controlled missile to search for opponents or explore the map

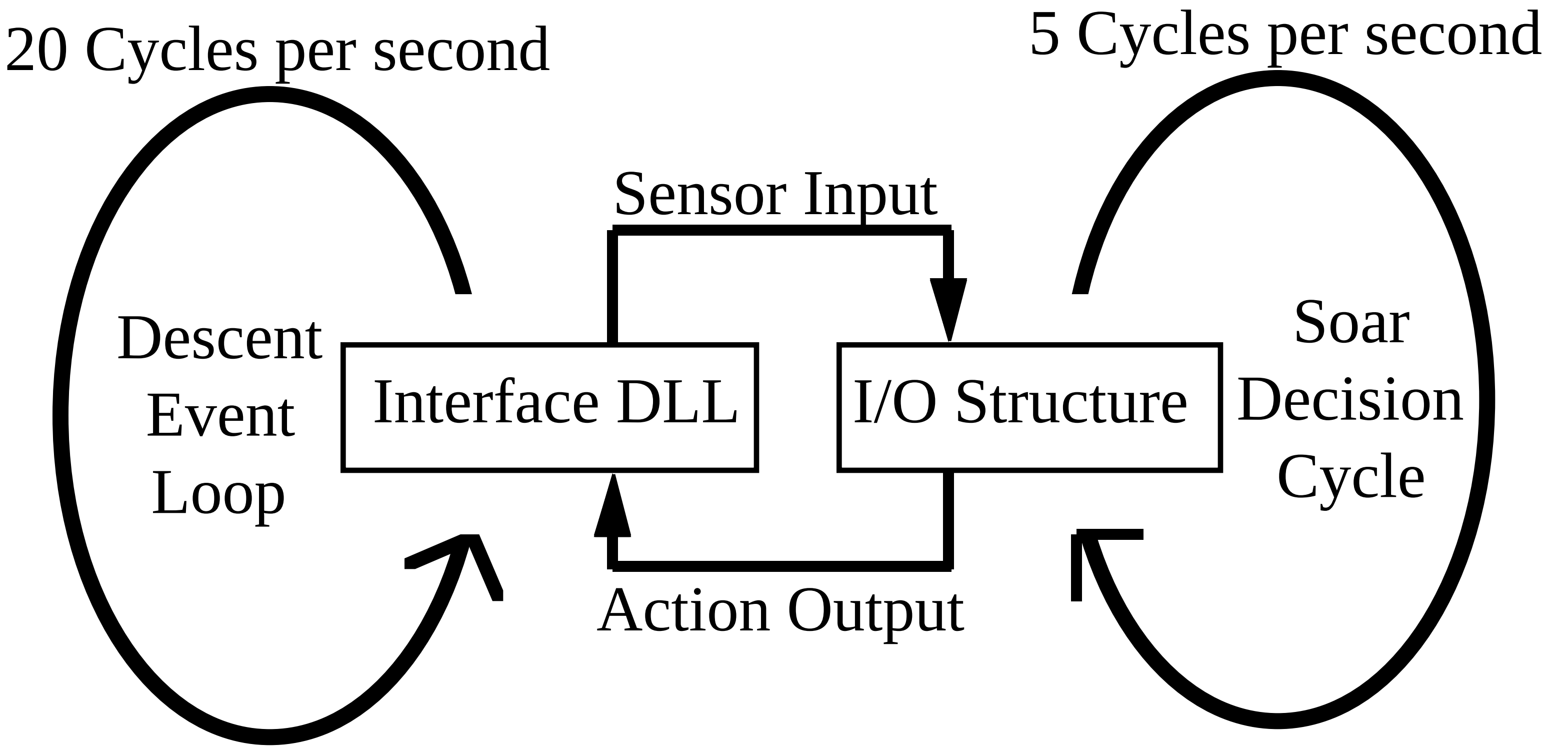


Programming Details

- Descent III will run under Windows 95 (maybe NT)
- Soar as a separate thread
  - Like multiprocessing in Unix
  - Separate event loops
  - Dual Processors?
- Interface as a Dynamically Loadable Library (DLL)
  - Soar compiles independently
  - Similar to the Soar/TclTk interface in Soar 7.1



Interface Diagram



20 Cycles per second

5 Cycles per second

Sensor Input

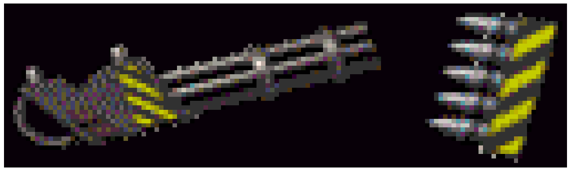
Descent Event Loop

Interface DLL

I/O Structure

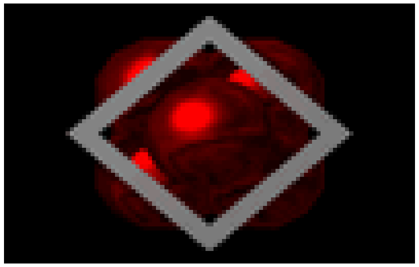
Soar Decision Cycle

Action Output



Input/Output Link

- Input Link
  - Ship Status
    - Available weapons and ammunition, Shields, Energy remaining
  - Map Information
    - Linked list of convex “sectors”
    - Path information from sector to sector and contents of each sector
  - Visual Sensor
    - Relative information about each entity (enemy, door, powerup...)
- Output Link
  - 23 Controls
  - Polled output style



Nuggets and Coal

- Nuggets
  - Many new challenges
    - Super fast reactions, New IO styles, New types of tasks
  - Lots of potential for future applications
    - Quake II, Abuse, other types of games
  - Impressive demos
- Coal
  - Lots of work
    - Programming interface, Programming Soar agent
  - Will Soar be fast enough?