Planning for Non-player Characters Using HTN and Visual Perception

M.Sc. Ibrahim Mahmoud
Prof. Dr.-Ing. M. Eng. Dieter Wloka

06, October 2015
Outline of talk

- Motivation
- NPCs Visual Perception
- Hierarchical Task Network Planning
- Strategic Planning Using HTN
- Scenario
  - Domain knowledge
  - Planning
- Experiment
  - Game
  - Issues
- Future Work Directions
  - New scenarios
  - Dynamic environments.
Motivation

- Serious games are designed to provide a learning platform – reduce costs, efforts and time spent in real-world training
- Cooperations and interactions in serious games play critical roles in training against constraints from the problem domain
- Believable non-player characters (NPCs) provide more realistic game experience and training environment for serious games
- Players can behave, react, interact and study with the designed believable NPCs while playing and training.
Motivation
Motivation

• Sense the environment:  
  – See, hear and smell

• Collect information: Memorize critical objects (short-term memory)

• Domain knowledge: (long-term memory)
  – Professional skills
  – Methods or action sequences to solve tasks

• Problem Solving: (strategic planning)
  – Generate plans to solve goal tasks

• Act:
  – Animate and interact with the environment as well as the player properly
NPCs Visual Perception

- Sensing the virtual environment like a human:
  - Limited view angle
  - See partially occluded objects
  - Not to sense occluded objects
  - Mimic center of attention
  - Smaller far objects will have higher probability to be missed
• Vision is the way NPCs perceive the environment
NPCs Visual Perception

- Camera mimics human eye using multi-level of concave retinas

- NPC “sees” an object when the ray hits the object

- Objects placed out of the view_distance can not be seen

- Each NPC has its own list of “seen” objects.
NPCs Visual Perception

- Implementing center of attention
  - Reducing the size of squares in this area by generating more rays
  - Probability to miss smaller object is reduced
  - Computationally more expensive
NPCs Visual Perception

- Angle and distance test
- Objects with different sizes
- A list of seen objects

**TABLE I. PERCENTAGE OF SEEN OBJECTS**

<table>
<thead>
<tr>
<th>Size, Location</th>
<th>Number</th>
<th>[0 - 60]</th>
<th>[60 - 90]</th>
<th>[90 - 130]</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>big, far</td>
<td>30</td>
<td>90%</td>
<td>90%</td>
<td>83%</td>
<td>87.6%</td>
</tr>
<tr>
<td>big, close</td>
<td>30</td>
<td>100%</td>
<td>100%</td>
<td>93.3%</td>
<td>97.7%</td>
</tr>
<tr>
<td>small, far</td>
<td>50</td>
<td>82%</td>
<td>78%</td>
<td>42%</td>
<td>67.3</td>
</tr>
<tr>
<td>small, close</td>
<td>50</td>
<td>96%</td>
<td>86%</td>
<td>66%</td>
<td>82.6%</td>
</tr>
</tbody>
</table>
Hierarchal Task Network Planning

Diagram showing the hierarchical task network for building a house, decomposing tasks into smaller sub-tasks.
Hierarchal Task Network Planning
Hierarchal Task Network Planning

• Collection of Tasks need to be fulfilled

• Tested against constraints from the problem domain

• Tasks are given in the form of network

• Tasks that satisfies its preconditions will be executed

• Primitive vs. non-primitive (compound) tasks.
• **Methods** are used for decomposing non-primitive tasks

• **Operators** fulfill the primitive tasks

• To decompose non-primitive tasks, methods can provide several **branches**

• Each one of these branches composes a sequence of actions
Hierarchical Task Network Planning

• Recursively executing the methods, until no farther decomposition is needed

• A final plan is found by the HTN planner if there exists a linear sequence of operators which could accomplish a given task

• C# Unity3D HTN-Planner based on the famous Dana Nau's Pyhop implementation which is written originally in Python
Strategic Planning Using HTN

- World Model
- Long-term Memory
- Short-term Memory
- State Manager
- Interface
- Vision Perception
  - Sense the task related objects
- Game World Environment
  - Task Trigger
  - Physical Body (animations)
- Planner
  - HTN
- Seek Plan
- Domain
Strategic Planning Using HTN

- World Model
- Long-term Memory
- Vision Perception
- Short-term Memory
- Primitive Actions
- Game World Environment
  - Task Trigger
  - Physical Body (animations)

Store information
E.g. Car:
  - Car_location (Transform)
  - Is_on_fire (Boolean)
  - Number_of_people_inside (Integer)

HTN
  - Planner
  - Seek Plan
  - Domain
Strategic Planning Using HTN

- World Model
- State Manager
- Vision Perception
- Primitive Actions
- Controller
- Game World Environment
  - Task Trigger
  - Physical Body (animations)

Long-term Memory

Initialize the main task the NPC will have to solve:
  e.g. HandleCarsAccident

Planner
Seek Plan
Domain

HTN
Information from memory represented in a way NPCs will understand

Update states

Strategic Planning Using HTN
Strategic Planning Using HTN

World Model

Long-term Memory

Short-term Memory

Vision Perception

Primitive Actions

Game World Environment

State Manager

Interface

Bridging the world model, planner and controller

Planner

HTN

Seek Plan

Domain

Task Trigger

Physical Body (animations)
Strategic Planning Using HTN

Planner will generate an action sequence to fulfill the goal tasks

Game World Environment
- Task Trigger
- Physical Body (animations)
Strategic Planning Using HTN

Controller contains real actions which control the NPCs in the game and plays the appropriate animations.
Method decomposition in HTN planner

Strategic Planning Using HTN
Strategic Planning Using HTN

• Method decomposition in HTN planner

Method: Extinguisher-fighting(fire)

- Goto-FireExtinguisher

- PickUp-FireExtinguisher

  - Goto-Fire

  - Fighting(fire)
Strategic Planning Using HTN

- Method decomposition in HTN planner

```
Method: Extinguisher-fighting(fire)

  Goto-FireExtinguisher

  PickUp-FireExtinguisher

    Goto-Fire

    Fighting(fire)
```
Method decomposition in HTN planner
Scenario

- Domain knowledge is extracted from (Das FeuerwehrLehrbuch).
- KATIE
- Unity3D game engine
- Virtual Environment: Burgstadt
Experiment

- Fighting fire in a building and rescuing victims
- **Nine** firefighters
  - Team leader
  - Detector
  - Mechanic
  - 2 water troop
  - 2 hose troop
  - 2 attack troop
Experiment

- **Sample of Results**

Team leader:
<go out of FF car>
  {Open Team Leader door}
  {Step out of FF car}
  {Close Team Leader door}
<Investigate the location>
  {Go to Pos1}
  {Go to Pos2}
- if all Firefighters gathered:
  {Give order to start fighting the fire}
  {Go to Pos3}

- Attack Troop Leader:
  - if WaterTroopLeader is out:
    {Step out of FF car}
    <Gather with FF group>
    {Go to Pos1}
  <Get Handlamp from FF car>
    {Go to Pos2}
    - if door closed:
      {Open door}
      {Get lamp in right hand}
  <Get verteiler>
    {Go to Pos3}
    - if door closed:
      {Open Door}
      {Get verteiler in left hand}
  <Place verteiler>
    {Go to Pos4}
    {Place the verteiler}
  <FightFire>
    {Go to Pos5} // hose head
    {Move the rope}
    - if water is ready:
      {Fight fire using water}
Experiment
Experiment

• Issues
  – Animation is not accurate and takes time
  – Ropes are computationally expensive, and keep moving (physics issue)
  – Vision is expensive
  – Changes in the scenario that should reflect in a change in the plan.
Future Work

- More scenarios, complicated scenes

- Human testing to assess believability of our NPCs behavior

- Improve visual perception
  - Lighting effect and weather conditions (fog, smoke and rain)

- Dynamic environment controlled by the player
  - Random variables effect the environment
  - Player can add or remove components to the scene.
Thank you for listening

Questions?!