Managing Massive Multiplayer Online Games
SS 2017
Exercise Sheet 3: Conflict management and dead reckoning
Discussion: May 17th, 2017

In the following regard an abstract game in which the players are in a two-dimensional world. Each player \( p \) has a positive count of health points \( p.H \in \mathbb{N} \). A player \( p_i \) can perform the following actions in the game:

- \( Heal(p_2, n) \) increases the health points of a player \( p_2 \) by \( n \) points. If \( p_2.H + n > 100 \), \( p_2.H \) is set to 100.
- \( Attack(p_2, n) \) reduces the health points of a player \( p_2 \neq p_1 \) by \( n \) points. If \( n > p_2.H \) player \( p_2 \) is dead and can not perform actions any more.

Exercise 3-1 Conflicts

Regard an instance of the game in which the following action requests are sent. Initially all three players have 50 health points, meaning \( \forall 1 \leq i \leq 3 : p_i.H = 50 \) The game uses a client-server architecture with central time processing for communication, i.e. the order of execution is determined by the server. With regard to the simplicity we assume that the latency is two ticks both for the transmission of an action to the server and for the transmission of an update from the server to the client.

<table>
<thead>
<tr>
<th>Player</th>
<th>Action</th>
<th>Time (Client)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_2 )</td>
<td>Attack( (p_1,60) )</td>
<td>1</td>
</tr>
<tr>
<td>( p_1 )</td>
<td>Attack( (p_2,30) )</td>
<td>2</td>
</tr>
<tr>
<td>( p_1 )</td>
<td>Heal( (p_1, 80) )</td>
<td>3</td>
</tr>
<tr>
<td>( p_2 )</td>
<td>Heal( (p_2, 60) )</td>
<td>4</td>
</tr>
<tr>
<td>( p_2 )</td>
<td>Attack( (p_3,30) )</td>
<td>5</td>
</tr>
<tr>
<td>( p_3 )</td>
<td>Attack( (p_2,50) )</td>
<td>6</td>
</tr>
<tr>
<td>( p_2 )</td>
<td>Attack( (p_3,30) )</td>
<td>7</td>
</tr>
</tbody>
</table>

To solve conflicts the approach \textit{resetting local actions} shall be used.

(a) How does the game proceed on the side of the server?

(b) How does the game proceed on the side of the client of player \( p_1 \)? Which anomalies occur?

(c) How does the game proceed on the side of the client of player \( p_2 \)? Which anomalies occur?

(d) How does the game proceed on the side of the client of player \( p_3 \)? Which anomalies occur?

(e) Which anomalies would be prevented locally for player \( p_3 \), if the clients were communicating via peer2peer and used a lag-mechanism with four ticks delay to solve conflicts? Assume a latency of two ticks for the communication between two clients.

(f) Discuss the advantages and disadvantages of these solutions!
Exercise 3-2  Dead Reckoning

To save bandwidth positions of players are not transmitted with every tick. Regard the client of player $p_1$ who perceives actions of another player $p_2$. The client of $p_1$ receives the following position updates of player $p_2$ from the server:

<table>
<thead>
<tr>
<th>Player</th>
<th>x</th>
<th>y</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_2$</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>$p_2$</td>
<td>110</td>
<td>90</td>
<td>15</td>
</tr>
<tr>
<td>$p_2$</td>
<td>130</td>
<td>90</td>
<td>30</td>
</tr>
<tr>
<td>$p_2$</td>
<td>160</td>
<td>50</td>
<td>40</td>
</tr>
</tbody>
</table>

At which position is player $p_2$ displayed at time 45? Use the following prediction models:

(a) The last known position is used as prediction as it is.

(b) To predict the position a linear movement with constant velocity is assumed.

(c) To predict the position a linear movement with constant acceleration is assumed.

Exercise 3-3  Hermite-Interpolation

The following situation of an abstract game on a two-dimensional field is given: The position of a player and his direction of movement at time $t$ are given by dead reckoning with a position vector $p_{DR}$ and a movement vector $d_{DR}$. At the same time an update from the server arrives with the real position vector and movement vector $p_{EX}, d_{EX}$.

Now the client has to transfer position and movement which were calculated with dead reckoning to the actual data within a time window $\delta$. On account of simplicity one can assume that within a time window $\delta$ a player moves exactly by the length of a movement vector. In other words, at time $t + \Delta t$ the player should be at position $p_{EX} + d_{EX}$.

The following vectors are given:

\[
p_{DR} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad d_{DR} = \begin{pmatrix} 2/3 \\ 3/2 \end{pmatrix} \quad p_{EX} = \begin{pmatrix} 4/2 \\ 2/1 \end{pmatrix} \quad d_{EX} = \begin{pmatrix} -1/1 \\ 1 \end{pmatrix}
\]

Illustrate the idea of position correction with linear combination of Hermite-functions as described in the script (chapter 3, page 20). For that calculate the value of the linear combination function $\hat{p}(x)$ (see below) for $x \in \{1/2, 7/8\}$. Mark these points in the plot and sketch your idea of the corresponding connecting curve based on these.

\[
h_1(x) = 2x^3 - 3x^2 + 1 \quad h_2(x) = -2x^3 + 3x^2 \\
h_3(x) = x^3 - 2x^2 + x \quad h_4(x) = x^3 - x^2
\]

\[
\hat{p}(x) = p_{DR} \cdot h_1(x) + (p_{EX} + d_{EX}) \cdot h_2(x) + d_{DR} \cdot h_3(x) + d_{EX} \cdot h_4(x)
\]

where $x \in [0, 1]$ describes the progress of movement between time $t$ and time $t + \Delta t$. 

2