Server/Client Architecture

To facilitate the development of AI agents for playing Angry Birds we have separated the game interaction functionality from the agent logic and reasoning functionality through a client/server architecture. The server communicates with the Angry Birds game and maintains a history of shots played by each agent. It exposes a simple communication protocol for the client to execute shots and perform auxiliary functions such as reload a level. This document described that protocol. The same protocol will be used for the Angry Birds AI Competition (ECAI 2014).

Changes from the IJCAI 2013 version:

The signal byte of termination is changed from "2" to "-1" (see section Termination)

A few commands are added to the server software (see section Run the Server)

Protocols

A typical client message contains a message header that shows the ID of the message (MID) and a message body (the body can be empty). All the messages (bytes) are encoded in network order (big endian) form.

Configuration Messages

A configuration message is sent by the client when it first connects with the server. The message contains the team ID (a four- bytes Integer). On receiving the message, the server will check the database to verify the ID. The client will be rejected if the ID is invalid. **Remember to get your unique ID from the organizers.**

The Message Format

MID	Team_ID

Return:

Round Info	Time_limit	**Number of Levels**

Round Info: a byte indicates the ongoing round of the competition.

1: 1st qualification round

2: 2nd qualification round

3: Group round (group of 4)

4: Knock-out round (group of 2)

0: Reject

Time_limit: a byte indicates the time limit in minutes

Number of Levels: a byte indicates the number of available levels.

In development version, the server will treat all Integers as valid ID.

In-Game Action Messages:

In the game, an agent is allowed to make shots and zoom out. There are two types of shots

1. Shots specified in the Cartesian coordinate system
2. Shots specified in the Polar coordinate system.

Note that shots will only be executed when in play mode and only in a certain region (see the following figure). This is done to prevent accidental clicking on buttons or the menu.

The server can execute a shot in the following two modes:

1. **Safe mode:**
   The server will wait 15 seconds after making a shot, and then record the final score when a won page shows up.
2. **Fast mode:**
   The server will send back a confirmation once a shot is made. The server will not do any check for the appearance of the won page.

✧ **Why we need the safe mode:**

It takes time for the objects affected by a shot to be stationary. Once all the objects are settled down, a won page will show up when the level is cleared, and then we can get the final score. This can take up to 15 seconds.

Agents will miss their scores if they send load/reload level requests before the won page is shown. So by enforcing a certain waiting time at the server side, we can ensure the score is correctly captured.

- **The server will try to record the score when**
  1. **It receives a loadLevel request**
  2. **It receives a restartLevel request**
  3. **It executed a shot in the safe mode.**

  So, if you submit shots in the Fast mode, you are at risk of missing the scores.

Recommendations:

1. **If you just want to make some trial shots, then use the Fast mode and you could get the response immediately**
2. **If you know you are about to clear a level e.g. there is only one pig left without sheltering, then you might want to submit a safe shot**
3. **If you use fast shots, we recommend that you analyse the screen shots yourself and only issue a loadlevel or restartlevel request once the won page shows a static score. Otherwise we might not record your score correctly.**

Shot in Cartesian coordinates (cshoot/cFastshoot)

The body of the message contains 6 parameters with each of the parameter consumes 4 bytes. The parameters are:

1. focus_x : the x coordinate of the focus point
2. focus_y: the y coordinate of the focus point
3. dx: the x coordinate of the release point minus focus_x
4. dy: the y coordinate of the release point minus focus_y
5. t1: the release time
6. t2: the gap between the release time and the tap time.

If t1 is set to 0, the server will execute the shot immediately.

The Message Format

MID	focus_x	focus_y	dx	dy	t1	t2

Return 1 or 0

1: the shot has been made

0: the shot has been rejected

Note: cshoot and cFastshoot have different MIDs.

Shot in Polar coordinates (pshoot/pFashshoot)

This message is almost the same as the chost message except it uses *r* and *theta* instead of d1 and d2.

r: the radial coordinate

theta: the angular coordinate by degree from -90.00 to 90.00. The theta value is represented by an integer

E.g. theta = 8025 means the degree is 80.25

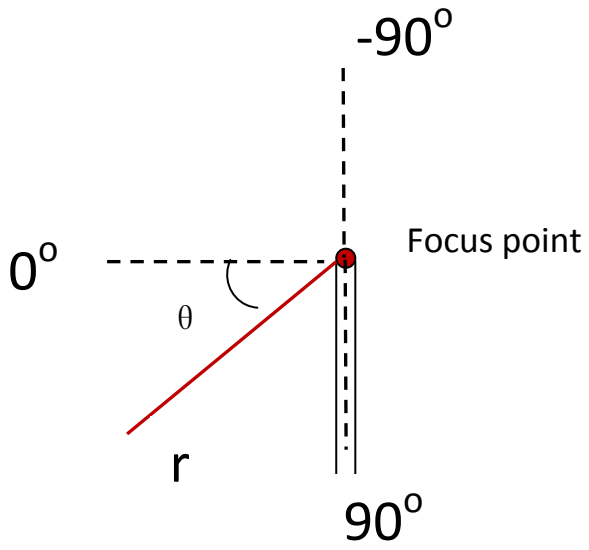
The Message Format

MID	focus_x	focus_y	r	theta	t1	t2

Return 1 or 0

1: the shot has been made

0: the shot has been rejected



Note: pshoot and pFastshoot have different MIDs

Shooting sequence

Shooting sequence contains a set of shots that will be made sequentially.

The Message Format (the shot below refers to either a cshot or pshot message):

MID	num of shots	shot	….	shot

Return:

An array with each slot indicates a good/bad shot. The bad shots are those shots that have been rejected.

For example, the server received 5 shots, and the third one was rejected due to some reason, then the server will return

[1][1][0][1][1]

You can also submit a shooting sequence in Fast mode. Then the server will send back a confirmation after all of the shots in the sequence have been executed.

FullyZoomOut

The server will fully zoom out on receiving this message.

The Message Format

MID

Return 1 or 0

1: The server has fully zoomed out in the current game.

0: The server cannot zoom out

When a level is loaded/reloaded, the server will perform the fully zoom out operation automatically (see section Level Selection Messages). The agents may want to use this command when they cannot detect the slingshot or “feel” the level is not fully zoomed out.

FullyZoomIn

The server will fully zoom in on receiving this message.

The Message Format

MID

Return 1 or 0

1: The server has fully zoomed in in the current game.

0: The server cannot zoom in

ClickInCenter

The server will make a click in the centre of the screen on receiving this message. You can use this message to move the camera of the game.

The Message Format

MID

Return 1 or 0

1: The server has made a centre click in the current game.

0: The server cannot make a centre click.

Query Messages

In the competition, an agent is allowed to query the current screen, game state, level and global scores.

Do Screenshot

The Message Format

MID

Return:

Width	Height	Image Bytes

The agent will be returned with a bytes array. The first and second four bytes tell the image width and height respectively. The remaining bytes are raw bytes (as RGB triples) of the image.

Get the State

The Message Format

MID

Return:

Ordinal of the state

The server will return One byte indicates the ordinal of the state

[0]: UNKNOWN [1] : MAIN_MENU [2]: EPISODE_MENU [3]: LEVEL_SELECTION [4]: LOADING [5]: PLAYING [6]: WON [7]: LOST

Get My Score

The Message Format

MID

Return:

Level 1 score	Level 2 score	……..	Level 21 Score

The server will return a fixed length (4 * 21) bytes array with every four slots indicates a best score of the corresponding level. Scores of unsolved and unavailable levels are zero.

Get the Current Level

The Message Format

MID

Return:

Level

Get Best Scores

The Message Format

MID

Return:

Level 1 score	Level 2 score	……..	Level 21 Score

The server will return a fixed length (4 * 21) bytes array with every four slots indicates a best score of the corresponding level. Scores of unsolved and unavailable levels are zero.

1: 1st qualification round: this request will return an array of the best scores the naive agent achieved on the qualification levels.

2: 2nd qualification round: this request will return an array of the best scores achieved in the 1st qualification round on the qualification levels.

3: Group round (group of four): this request will return an array of the current best scores of their own group. The scores change whenever a new high score is obtained.

4: Knock-out round (group of two): this request will return an array of the current best scores for the two participating agents. The scores change whenever a new high score is obtained.

Level Selection Messages

An agent is allowed to select a level by using the following two messages

The following are the operations performed by the server to load/reload a level:

1. Click on the corresponding buttons to go to the level
2. Fully zoom out when “PLAYING” state is detected
3. Return the confirmation

Load a Level

The Message Format

MID	Level

The level is indicated by one byte. The value can be set from 0 to the maximum level number (will be released before the competition). To load the current level, please set the value to 0.

Return:

Return 1 or 0

1: the level has been loaded

0: The server cannot load the level

Restart a Level

The Message Format

MID

Return 1 or 0

1: the level has been restarted

0: The server cannot restart the level

Termination:

There is a timer running at the server side. Once the time limit reaches, the server will turn into the “about to terminate” mode. In this mode, all the incoming requests will be responded with a signal byte [-1]. Once a client received the signal byte, it should begin to store all the data to the local disk. The server will drop all the clients 3 minutes after being in the “about to terminate” mode. At that time we will terminate all agents if they have not yet done so themselves.

Run the Server

1. First start the server from the command line

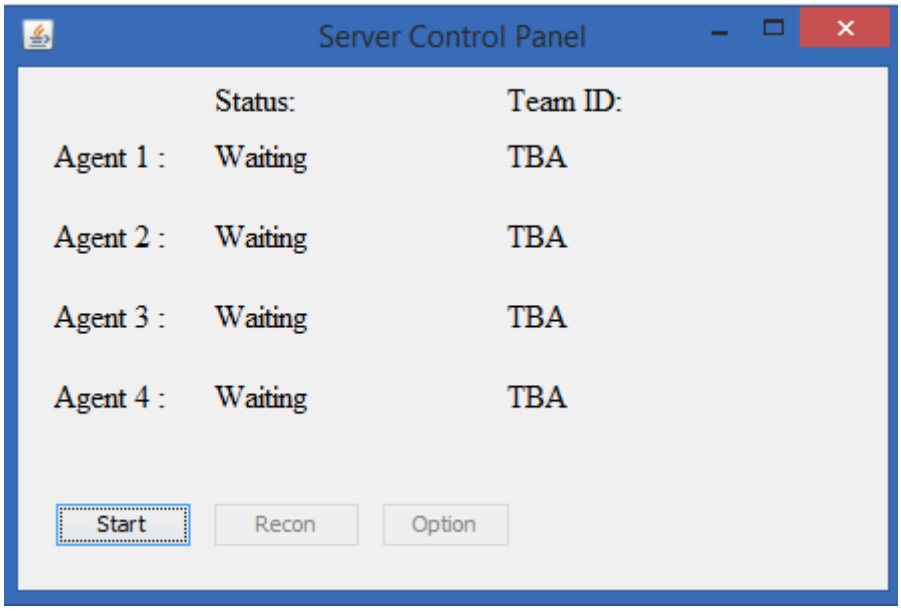
   java -jar ABServer.jar -t [time limit in minutes] -n [number of levels] -r[round info]

   E.g. *java -jar ABServer.jar -t 127 -n 10 -r 1* starts the server with 127 minutes as the time limit, 10 as the total number of levels, and 1st qualification round as the on-going round.

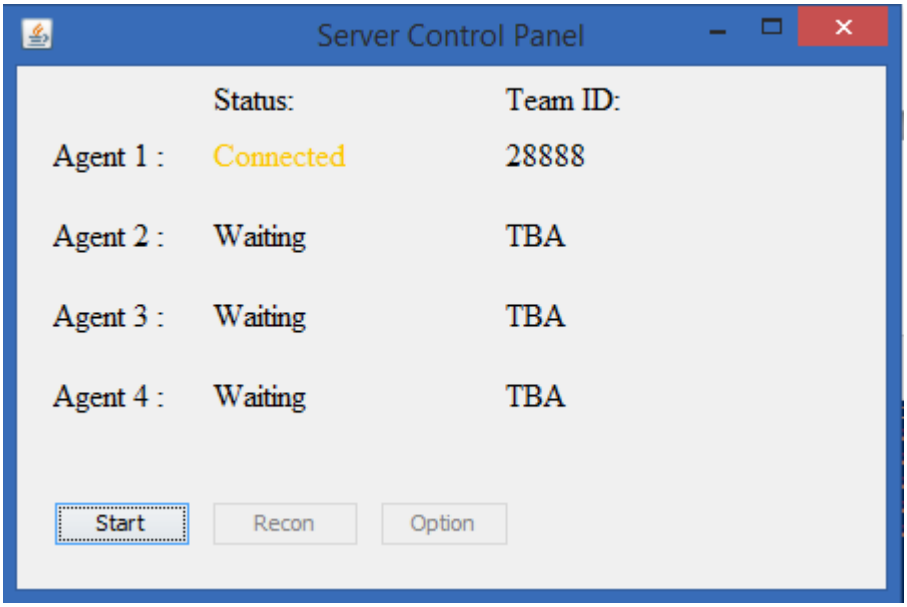
   java -jar ABServer.jar -t 0 starts the server without the time limit.

   java -jar ABServer.jar starts the server with the default setting (-t 0 -n 21 -r 1)

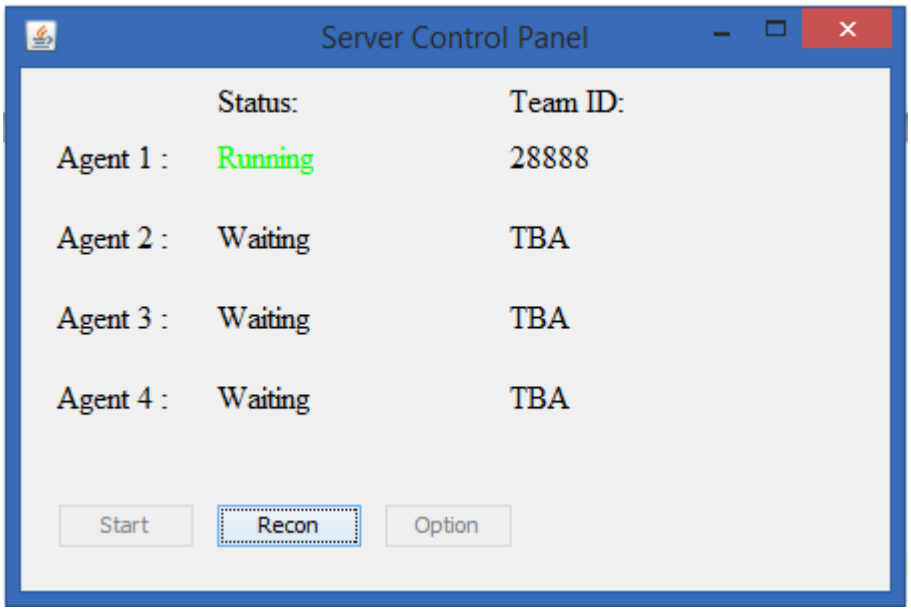
   Then you will see this window:



2. Then connect the client, once a client is connected, the connection status will become "Connected"



3. Press "Start" to start the server. The server will drop all clients when time is up.



When your agent is disconnected during the play (before server terminates), you can reconnect your agent by clicking the button “Recon”

Summary of the Protocols

MID	Request	Format (byte[])	Return	Format (byte[])
1-10	Configuration Messages			
1	Configure (Team ID)	[1][ID] ID: 4 bytes	Four bytes array. The first byte indicates the round; the second specifies the time limit in minutes, and the third specifies the number of available levels	[round info][time limit][available levels] Round info: [1]: 1st qualification round [2]: 2nd qualification round [3]: group round [4]: knock-out round [0:] Reject
11-30	Query Messages			
11	Do screenshot	[11]	Width, height, plus image bytes	[width][height][image bytes] width, height: 4 bytes
12	Get the state	[12]	One byte indicates the ordinal of the state	[0]: UNKNOWN [1] : MAIN_MENU [2]: EPISODE_MENU [3]: LEVEL_SELECTION [4]: LOADING [5]: PLAYING [6]: WON [7]: LOST
13	Get Best Scores	[13]	A fixed length (21 * 4bytes)bytes array with every four slots indicates a best score for the corresponding level	[score_level1]….[score_level21]
23	Get my score	[23]	A fixed length (21*4) bytes array with every four slots indicates a best score for the corresponding level	[score_level1]….[score_level21]
14	Get the current level	[14]	One byte indicates the current level	[1-21]
31-50	In-Game Action Messages			

31	Shoot using the Cartesian coordinates [Safe mode]	[31][fx][fy][dx][dy][t1][t2] Each parameter consumes 4 bytes	OK/ERR	[1]/[0]
41	Shoot using the Cartesian coordinates [Fast mode]	[35][fx][fy][dx][dy][t1][t2] Each parameter consumes 4 bytes	OK/ERR	[1]/[0]
32	Shoot using Polar coordinates [Safe mode]	[32][fx][fy][theta][r][t1][t2] Each parameter consumes 4 bytes	OK/ERR	[1]/[0]
42	Shoot using the Polar coordinates [Fast mode]	[36][fx][fy][theta][r][t1][t2] Each parameter consumes 4 bytes	OK/ERR	[1]/[0]
33	Sequence of shots [Safe mode]	[33][shots length][MID][Params][MID][Params] Maximum sequence length: 16 shots	An array with each slot indicates good/bad shot. The bad shots are those shots that are rejected by the server	For example, the server received 5 shots, and the third one was not executed due to some reason, then the server will return [1][1][0][1][1]
43	Sequence of shots [Fast mode]	[37][shots length][MID][Params][MID][Params] Maximum sequence length: 16 shots	An array with each slot indicates good/bad shot. The bad shots are those shots that are rejected by the server	For example, the server received 5 shots, and the third one was not executed due to some reason, then the server will return [1][1][0][1][1]
34	Fully Zoom Out	[34]	OK/ERR	[1]/[0]
35	Fully Zoom In	[35]	OK/ERR	[1]/[0]
36	Click In Center	[36]	OK/ERR	[1]/[0]
51-60	Level Selection Messages			
51	Load a Level	[51][Level] Level : 1 byte	OK/ERR	[1]/[0]
52	Restart a level	[52]	OK/ERR	[1]/[0]

The server will respond all the requests with a signal byte [-1] when time is up