Cricket Tracker Quick Start Guide

March 2006

Software installation

Requirements

- J2SE Java Runtime Environment (JRE), version 1.4.2 or higher
  download from: http://java.sun.com/j2se/1.4.2/download.html
- RXTX gnu.io package, version 2.1-7 (final)
  download from: ftp://ftp.qbang.org/pub/rxtx/rxtx-2.1-7-bins-r2.zip or http://rxtx.org
- Cricket Tracker Server and Client Applet (version 0.3)
  both are contained in one Java archive file: CricketTracker.jar

Procedure

1. Install the Java Runtime Environment, follow the instructions from the installer.
2. Install the RXTX package, follow instructions in the file 'Install' in the downloaded zip file. OS-X users: the installer doesn't work, you need to manually create folder /var/lock, chgrp to uucp, chmod to 775, and make yourself member of group uucp. Like this:
   ```
   sudo mkdir /var/lock;chgrp uucp /var/lock;chmod 755 /var/lock
   sudo niutils –mergeprop / /groups/uucp users `whoami`
   ```
3. Test the installation by opening a console window, changing directory to the directory containing the CricketTracker.jar file, and typing:
   ```
   java –jar CricketTracker.jar
   ```

Hardware installation

Requirements

- A Mac/Windows/Linux computer with at least 3 free serial ports, either build-in or from a USB-to-Serial converter, like the Keyspan USA-49WLC 4-Port USB-Serial adapter.
- At least 4 Crickets, programmed with firmware version 2.3.1 or later.
  See http://cricket.csail.mit.edu for Cricket documentation.
- Serial cables for each Cricket which is to function as a listeners (at least 3 units, connected to the PC).
- Optional 5V power supply to power the Cricket listeners, so they don't have to run on batteries, and can be switched on/off remotely.

Procedure

1. Connect each Cricket in turn to a serial port, and use a terminal emulator to connect to the Cricket. Crickets communicate at 115.200 baud, 8 bits, no parity, no handshake.
2. Verify the connection by typing the command: `G CF<return>`
   The Cricket should respond with a configuration overview.
3. Program each Cricket as a Beacon by issuing the command: `P MD 1<return>`
4. Save the state in Flash memory with the command: `P SV<return>`
5. Connect at least 3 Crickets to the PC, at least one remaining Cricket will be used as wireless beacon. The Crickets which are connected to the PC will be temporarily configured as listeners by the Cricket Tracker application.
6. Make sure switch SW1 of each Cricket connected to the PC is NOT in the TEST position.

Starting the Cricket Tracker server

The Cricket Tracker graphical user interface (GUI) is started from a console window by typing:
`java –jar CricketTracker.jar`

Alternatively, the tracker can be started without a GUI by typing:
`java –jar CricketTracker.jar –nogui`

The response in the console will be similar to:

```
Server started on 192.168.0.2:12000
COM1 opened
COM2 opened
COM3 opened
COM4 opened
```

If the GUI is started, the embedded client applet in the GUI will connect to the server, resulting in a message similar to:

```
Client 127.0.0.1:3270 connected
XML request from 127.0.0.1:3270
```

For each Cricket listener connected through the serial ports, a message will be displayed (when the Crickets are switched on) showing the id of the Cricket:

```
Listener 01:8f:aa:60:0a:00:00:87 connected to COM1
Listener 01:ab:03:61:0a:00:00:a2 connected to COM2
Listener 01:1a:a7:60:0a:00:00:31 connected to COM3
```

When a wireless Cricket Beacon is in reach of one or more Listeners, the Listeners send messages to the Tracker whenever a distance measurement is made. These messages are displayed in the console window, showing a timestamp (t) the id of the Listener (src), and the Beacon data:

```
BeaconMessage { t=1121038650618, src=01:8f:aa:60:0a:00:00:87,
data={DB=110, ID=01:1c:61:61:0a:00:00:20} }

Beacon 01:1c:61:61:0a:00:00:20 entered

BeaconMessage { t=1121038650618, src=01:1a:a7:60:0a:00:00:31,
data={DB=103, ID=01:1c:61:61:0a:00:00:20} }

BeaconMessage { t=1121038650607, src=01:ab:03:61:0a:00:00:a2,
data={DB=109, ID=01:1c:61:61:0a:00:00:20} }
```

After three readings from the same Beacon by different Listeners within a short time frame are received, the 3d position of the Beacon is calculated and sent to the connected clients:

```
BeaconPosition 01:1c:61:61:0a:00:00:20 { 56.36, 33.02275, 87.258514 }
```

OSC support

The Tracker can send OSC messages to any number of OSC servers, such as MAX/MSP or SuperCollider. To open a connection to a OSC server, add `-osc host:port` to the command line for each connection. The default host is localhost, so just a port number can be specified to connect to a OSC server running on the same machine.

Supported OSC messages are:

`/cricket/beacon/entered [string]id`
Sent when a beacon is dectected by one or more listeners.

`/cricket/beacon/moved [string]id [int]x [int]y [int]z`
Sent when a new 3-d beacon position is calculated.

`/cricket/beacon/lost [string]id`
Sent when a new beacon position could not be calculated since 5 seconds. The beacon is however still detected as being present in the space.

`/cricket/beacon/gone [string]id`
No signal has been received from a beacon since 5 seconds.

`/cricket/listener/entered [string]id`
Sent when a listener connected to one of the comm ports.

`/cricket/listener/moved [string]id [int]x [int]y [int]z`
Sent when the coordinates of a listener have changed (manually, in the GUI).

`/cricket/listener/gone [string]id`
Sent when a listener is no longer connected.

The Graphical User Interface

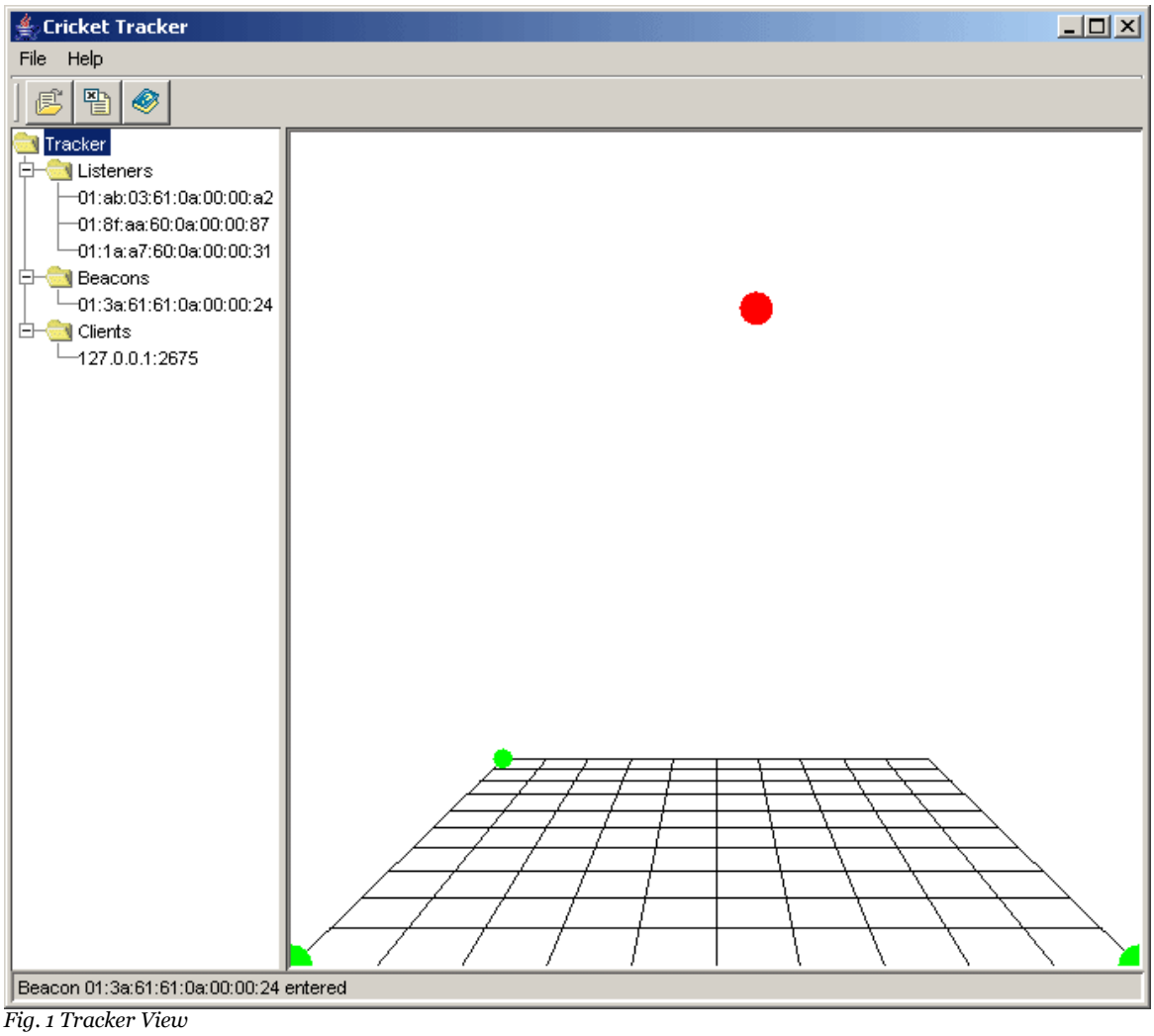

Fig. 1 Tracker View

Fig. 1 shows the default view of the graphical user interface. The left pane contains a tree representing the objects connected to the tracker. Crickets which are physically connected to the PC operate as Listeners, and are listed by their unique id in the Listeners folder.
Wireless Crickets which are in reach of the Listeners are listed by their unique id in the Beacons folder.
Remote clients, connected by TCP/IP over a local or wide area network, are listed by their IP number and remote port number in the clients folder. The first client is the embedded client applet which is visible in the right pane.
The client applet shows a 3d perspective view of the reference plane, containing the Listeners, identified by a green dot. Each wireless Beacon is identified by a red dot and placed in the perspective view according to its 3d position, as calculated by the Tracker.

Listener panel

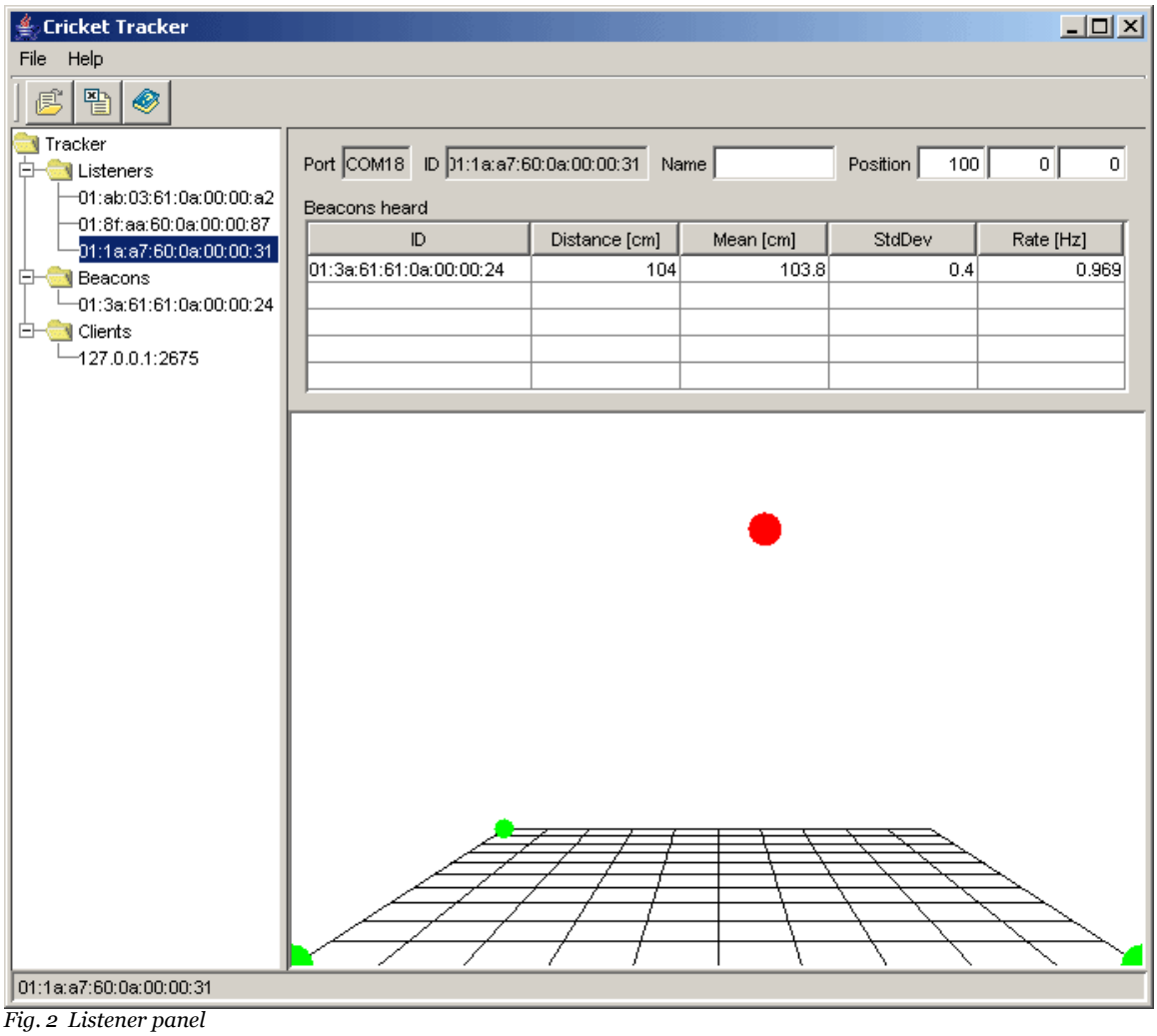

Fig. 2 Listener panel

When one of the Listener nodes in the tree is selected, a Listener panel is shown (see Fig. 2), containing data for that Listener: the name of the serial port the Listener is connected to, the Listener's unique id, an optional name, and its coordinates.

The table lists all Beacons which are heard by this Listener. For each Beacon, the last measured distance, the mean distance (of the last 5 readings), its standard deviation, and the update rate are shown in real-time.

The Listener's name and coordinates can be modified directly in this panel, and are saved in the system preferences, and retrieved whenever this Listener (recognized by its unique id) connects to the Tracker. Note: you have to hit Enter in each field for changes to be saved.

For the position calculation (by trilateration) to succeed, the z-coordinate of all Listeners should be zero, meaning all Listeners should either be on the floor or on the ceiling. One listener should be at the origin (0,0,0), one on the x-axis (y=0), and the other(s) can be anywhere on the z=0 plane.
In future versions, placement of Listeners and dimensions of the viewing volume will be more flexible.

Beacon panel

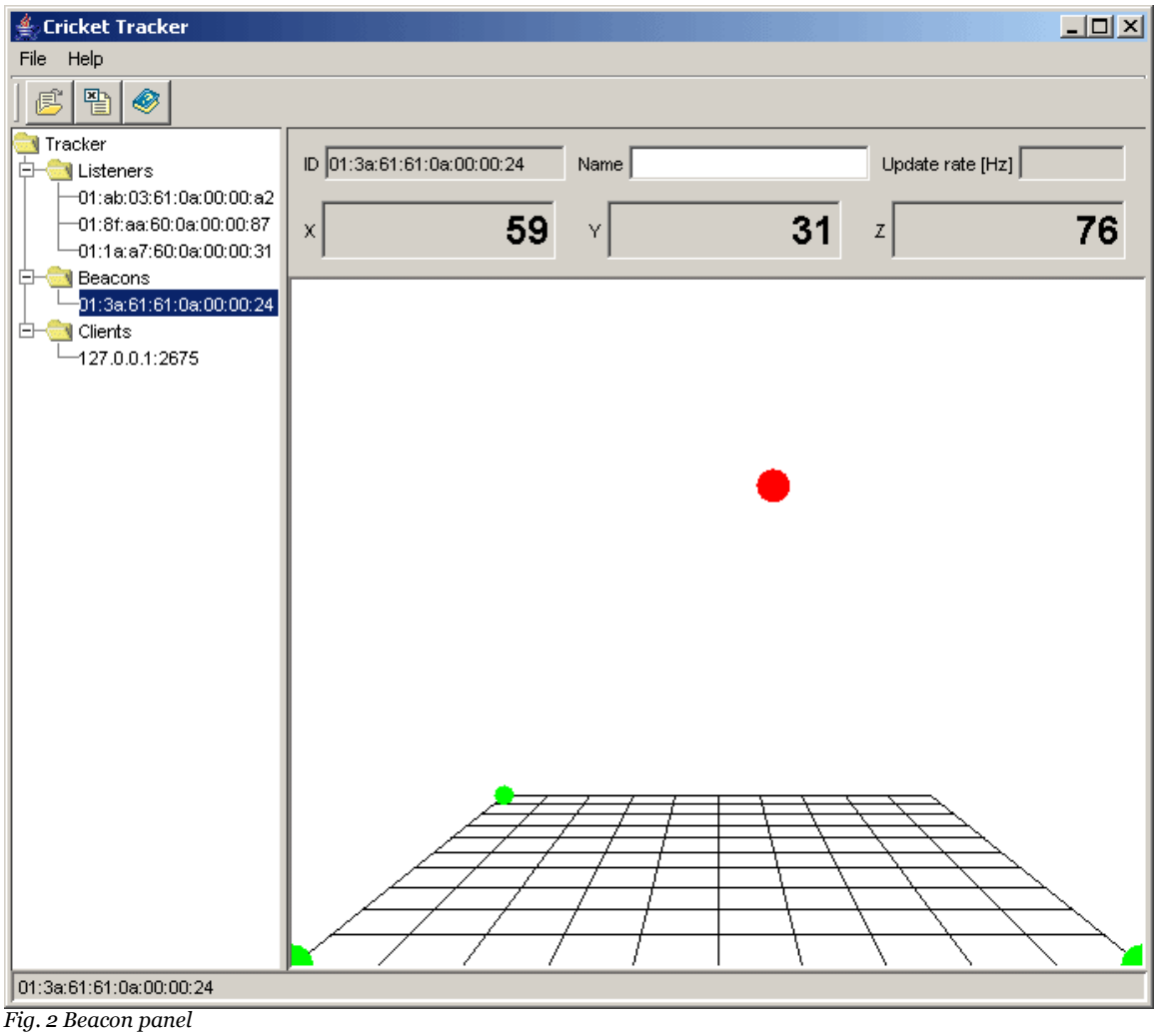

Fig. 2 Beacon panel

When a Beacon is selected, the Beacon panel opens, showing the Beacon's unique id, its name, the update rate and its coordinates in real-time (Fig. 3).

Configuration Files

The Cricket Tracker configuration is saved in the system preferences (the registry under Windows), but can be saved to and loaded from a file. The configuration files are in XML format. Configuration files can contain the port number of the server, a list of serial ports to be used (if not specified, all serial ports are attempted), and names and coordinates of Listeners.

A sample configuration file is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE preferences SYSTEM 'http://java.sun.com/dtd/preferences.dtd'>
<preferences EXTERNAL_XML_VERSION="1.0">
  <root type="system">
    <map />
    <node name="crickettracker">
      <map />
      <node name="01:1a:a7:60:0a:00:00:31">
        <map>
          <entry key="z" value="0.0" />
          <entry key="y" value="0.0" />
          <entry key="x" value="100.0" />
        </map>
      </node>
      <node name="01:8f:aa:60:0a:00:00:87">
        <map>
          <entry key="x" value="0.0" />
          <entry key="y" value="100.0" />
          <entry key="z" value="0.0" />
        </map>
      </node>
      <node name="01:ab:03:61:0a:00:00:a2">
        <map>
          <entry key="x" value="0.0" />
          <entry key="y" value="0.0" />
          <entry key="z" value="0.0" />
        </map>
      </node>
      <node name="ports">
        <map>
          <entry key="COM1" value="" />
          <entry key="COM2" value="" />
          <entry key="COM3" value="" />
          <entry key="COM4" value="" />
        </map>
      </node>
    </node>
  </root>
</preferences>
```