Simulating Humanoid Robots in the Cloud

the testing behind the biggest world competition

Jose Luis Rivero <jrivero@osrfoundation.org>
Open Source Robotics Foundation (OSRF)
www.osrfoundation.org
let’s travel in time... 2011
Fukushima disaster


Photo: this work by The Virtual Union
let’s travel in time a bit more... 2015
June, 2015. California

Pomona club raceway

June, 2015. California
The primary technical goal of the DRC is to develop human-supervised ground robots capable of executing complex tasks in dangerous, degraded, human-engineered environments.
let’s travel in time even more... 2013
Virtual Robotics Challenge

---        | ---       | ---       | ---
DRC call   | VRC       | DRC trials| DRC finals
Open Source Robotics Foundation
VRC: virtual robotics challenge
VRC: virtual robotics challenge

[Logos for Gazebo, ROS, and CloudSim]
Planning the contest: getting testers for free
Prior to VRC

Preparing VRC

v1

v2

v3

v4

Gazebo

Gazebo

Gazebo

Gazebo

DARPA Robotics Challenge

DARPA Robotics Challenge

Open Source Robotics Foundation
vrc_bytecounter overcounting certain messages

6/5/13

Dear DRCSim folks,

After some exhaustive hunting down, I have determined that vrc_bytecounter over-counts bytes in certain circumstances, because it relying on the (incorrect) assumption that the Ethernet contribution to a packet is always only 14 bytes. For small packets, the hardware layer may add a trailer to pad out the frame.

Executive summary: vrc_bytecounter needs to use the IPv4 total length field, not the total captured bytes. From here, it can subtract off the IP header (and protocol headers as appropriate).

See details below.

Thanks-
to DRCSim,

And I wrote a patch for you too...

---

byte-count-patch....
```c
  totalPackets++;
  totalLength += header->len;

-  /* Get ethernet header size */
-  total_header_length += SIZEETHERNET;

-  /* Get IP header size */
-  ip = (struct ip _header*) (packet + SIZEETHERNET);
-  size_ip = IP_HL(ip) * 4;

+  total_header_length += protocol_header_length;

+  // get ip header and change endianness (works for little endian only!)
+  int ip_len = (((ip->ip_len & 0x00FF) << 8) | ((ip->ip_len & 0xFF00)>> 8);
+  printf("ip_len: %d", ip_len);
+  
+  if (uploading)
+  { 
-   totalPacketsUploaded += (header->len - total_header_length);
+   totalPacketsUploaded += (ip_len - total_header_length);
+  }
+  else
+  { 
-   totalPacketsDownloaded += (header->len - total_header_length);
+   totalPacketsDownloaded += (ip_len - total_header_length);
+  }

/* Update Redis */
```
participation is great!!

power to participation
Power to participations

- Everyone is lazy. You are lazy. I’m super lazy.

- Gazebo is large piece of C++
  - More than 40 dependencies
  - More than 200K lines of code
  - Could take more than 1 hour to compile
send a patch
patch accepted
merged in code
nightly package generation

v2

GAZEBO

send a patch
patch accepted
merged in code

v3

GAZEBO

precious time

precious time

DARPA ROBOTICS CHALLENGE

Open Source Robotics Foundation
The million euro question:

Best way of simulating the whole contest?
The million euro answer:

Which is the best way of simulate the contest?

To run a contest.
running the contest without running the contest?

Organize a rehearsal or training session
Participants in the loop
The simulator: DRCSim
Continuous integration:
### Continuous Integration: Jenkins

<table>
<thead>
<tr>
<th>S</th>
<th>W</th>
<th>Name</th>
<th>Last Success</th>
<th>Last Failure</th>
<th>Last Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>drcsi-m-any-cattle ws-gazebo4-devel-precise-amd64</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>drcsi-m-any-cattle ws-gazebo4-devel-trusty-amd64</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>drcsi-m-deb builder</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>drcsi-default-devel-precise_hydro-amd64</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>drcsi-default-deb</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>drcsi-default-gazebo-default-devel-precise-amd64</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>drcsi-drcsi 2.2-gazebo-gazebo-1.5-devel-precise-amd64</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>drcsi-drcsi 2.2-gazebo-gazebo-1.5-devel-quantal-amd64</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Testing the simulator

- 3D Robotics simulator
  - rendering (integration tests)
  - integrated with Jenkins
Testing the simulator

- 3D Robotics simulator
  - rendering (integration tests)
  - integrated with Jenkins

- Server Fixture
  - GTest fixture
    - spawns a gzserver
  - Provides common operations
    - compare images
    - create sensors
Testing the simulator (contest features)

- Specific contest aspects
  - Tests for the **full duration** of contest
    - Hidden surprises: memory leaks, energy problems, heating, etc.
Testing the simulator (contest features)

- Specific contest aspects
  - Tests for the **full duration** of contest
    - Hidden surprises: memory leaks, energy problems, heating, etc.
  - Be sure to test scoring
    - standard and alternative/cheating cases
  - API based on network channels
    - specific tool to check ROS topic/services
Jenkins headless slave (GPU)

- No monitor machines
  - nvidia needs special config in xorg.conf
  - `nvidia-xconfig --allow-empty-initial-configuration`

- Auto login + script for permissions
  - lightdm to run a script on login
  - `display-setup-script=/etc/lightdm/xhost.sh`
    - xhost +si:localuser:jenkins
Jenkins headless slave (GPU)

- No monitor machines
  - nvidia needs special config in xorg.conf
  - `nvidia-xconfig --allow-empty-initial-configuration`

- Auto login + script for permissions
  - lightdm to run a script on login
  - `display-setup-script=/etc/lightdm/xhost.sh`
    - `xhost +si:localuser:jenkins`

- Support for different linux distributions
  - **Important:** Xorg stack needs to be the same in host and chroot
going to the cloud
user@local.machine$ ssh user@cloud.machine
...
user@ip-10-254-666:~$
user@local.machine$ ssh ...
user@ip-10-254-666:~$
user@local.machine$ ssh
...
user@ip-10-254-666:~$
user@local.machine$ ssh user@cloud.machine
...
user@ip-10-254-666:~$

> interaction with operative system

Created by Uwe Kils (iceberg) and User:Wiska Bodo (sky). - (Work by Uwe Kils) http://www.ecoscope.com/iceberg/
user@local.machine$ ssh user@cloud.machine
...
user@ip-10-254-666:~$
user@local.machine$ ssh user@cloud.machine
...
user@ip-10-254-666:~$
Cloud first experience
Cloud first experience

Local

Cloud
system clock

![Graph showing system clock with sample number on the x-axis and elapsed time (ms) on the y-axis.](image-url)
system clock

Sample number

Elapsed time (ms)

'./out_cg1_long:x' using 0:1

Open Source Robotics Foundation
The dragon was probably virtualized.
production systems in the loop
last minute security bug?
There is only one rule:

don’t change or write code
“What kind of world is this that can send machines to Mars and does nothing to stop the killing of a human being?”
Questions and beers here.