Combining Roborescue and XABSL

Final Presentation

Maarten de Waard

UvA

June 28, 2012

1 Recap

2 Challenges
   - VB and C++
   - JXABSL Engine

3 Approach
   - JXI

4 Results

5 Demo

6 Conclusion

Recap

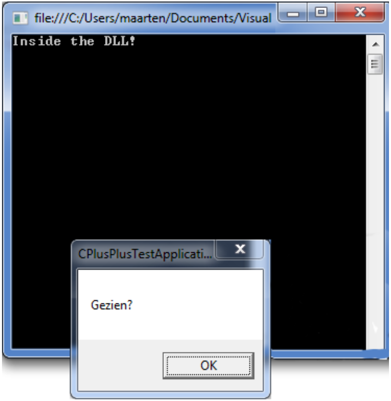
To freshen your memories, a short summary of my research and my goal:

- UsarCommander - the program used by the UvA Rescue team in the RoboCup
- XABSL - eXtended Agent Behavior Specification Language
- The combination - A thriving combination of UvA's Rescue research and Germany's winning robotic soccer team.

Visual Basic and C++

Problem: The XABSL Engine written in C++, UsarCommander in Visual Basic Solution: create a Dynamic Link Library (DLL) containing XABSL. Approach:

- Create runnable DLL, and run it from Visual Basic
- Create a C++ program implementing the needed XABSL files
- Create DLL from the C++ program.


file:///C:/Users/maarten/Documents/Visual
CPlusPlusTestApplicati...
OK

```
Inside the DLL!
```

Gezien?

Visual Basic and C++: The problems

- No experience: Creating a DLL from an entire framework is different than a 'Hello World' DLL.
- Cryptical errors

The error:

```
error LNK2019:  unresolved external symbol
"public:  void __thiscall
xabsl::Parameters::registerDecimal(char const
*,double &)"
(?registerDecimal@Parameters@xabsl@@QAEXPBDAAN@Z)
referenced in function "public:  virtual void
__thiscall TestBehavior::registerParameters(void)"
(?registerParameters@TestBehavior@@UAEXXZ)
```

JXABSL

Luckily, there was an alternative to the C++ engine:

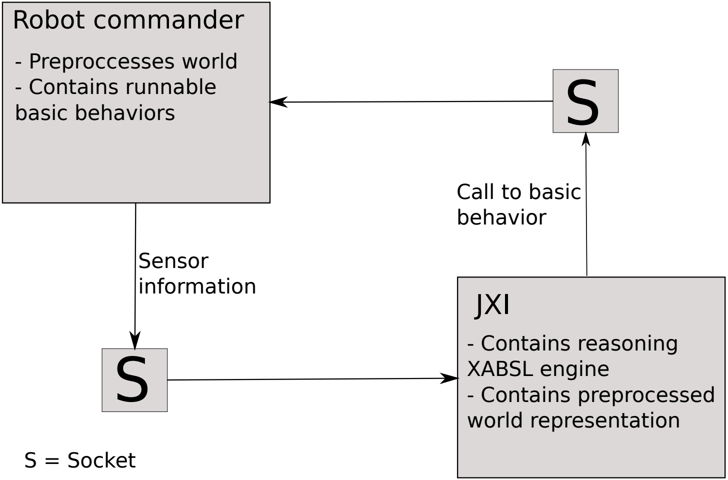
- XABSL Engine programmed in Java
- Little documentation
- Impossible to create direct connection to VB

Solution:
Socket connection between JXabsl and UsarCommander

JavaXabslImplementation

- Framework providing all XABSL-possibilities over a socket connection
- Easier to use than JXABSL
- Modular

Information flow



Robot commander

- Preproccesses world
- Contains runnable basic behaviors

Sensor information

S

Call to basic behavior

S

JXI

- Contains reasoning XABSL engine
- Contains preprocessed world representation

S = Socket

XABSL behavior

Two simple behaviors were created:

- Drive_circle:
  - Makes the robot autonomously drive a circle
- Walk_corridor
  - Makes the robot traverse though a corridor without bumping into anything

Code for walk_corridor

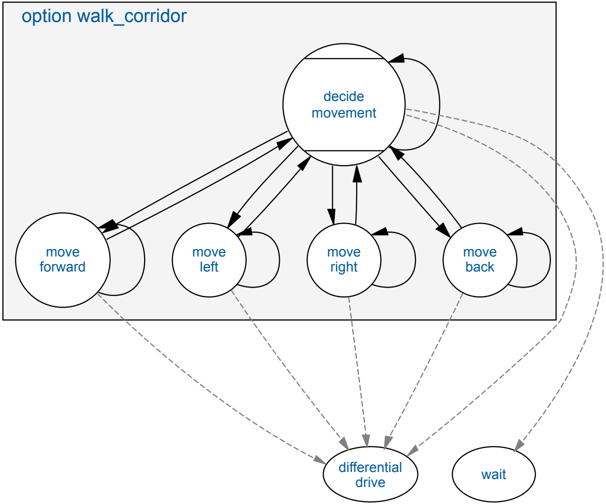
```
/** Uses the data from the laser sensor to walk through a
 * corridor as good as possible, without bumping into the walls */
option walk_corridor{
  initial state decide_movement{
    decision{
      if(laser_max < maximum_laser_value){
        goto move_back;
      }
      else{
        if(laser_max == laser_min_n){
          goto move_forward;
        }
        else{
          if(laser_max == laser_min_nne || laser_max ==laser_min_ne
                         || laser_max == laser_min_ene)
          {
            goto move_right;
          }
          else{
            if(laser_max == laser_min_nnw || laser_max ==laser_min_nw
                             || laser_max == laser_min_wnw)
            {
              goto move_left;
            }
            else{
              stay;
            }}}}}
    action{
      differential_drive(speed=0, turning_speed=0);
      wait(time=2);
    }
```

Code for walk_corridor

```
  state move_left{
    decision{
      /** go on till threat is over */
      if(laser_max == laser_min_nne || laser_max ==laser_min_ne
                || laser_max == laser_min_ene || laser_max == laser_min_n){
        goto decide_movement;
      }
      else{
        stay;
      }
    }
    action{
      differential_drive(speed=forward_speed, turning_speed=turning_speed);
    }
  }

  state move_back{
    decision{
      if(laser_max > maximum_laser_value){
          goto decide_movement;
      }
      else{
          stay;
      }
    }
    action{
      differential_drive(speed=-reverse_speed, turning_speed=0);
    }
  }
}
```

Generated option graph



Demo

A film of the robot turning 360 degrees, powered by JXI!

`http://www.youtube.com/watch?v=0ixdA-mzoCg&feature=youtu.be`

Conclusion

Good points:

- In theory, everything works.
- JXI can be combined with any program, because of its use of sockets
- The framework offers a lot of possibilities
- The framework offers easier understanding and implementing of XABSL

Possible improvements:

- The framework currently only works with one robot
- More complex behaviors could be implemented, to test the frameworks abilities
- Fixing the bugs in UsarCommander would seriously increase speed of the complete program.

Questions