Kinect Interface for UC-win/Road:

Application to Tele-operation of Small Robots

Hafid NINISS
Forum8 - Robot Development Team

Abstract: The purpose of this work is to develop a man-machine interface for the UC-win/Road VR software to improve the interaction between the real and virtual worlds. The Kinect plugin interface presented in this paper is based on 3D depth sensor integrating IR technology. The development started with the Kinect sensor, a device initially released by Microsoft as an input controller for its Xbox video game console, then moved to the Xtion Pro, a smaller similar sensor. This paper presents the Kinect Plugin as well as two applications oriented towards Robotics, to demonstrate the Kinect Plugin capabilities.

Key Words : Virtual Reality, Human-machine interface, Robot Tele-operation

1. Introduction

One of the main software solutions developed at Forum8 is UC-win/Road, a multi-purpose 3D-interactive Virtual Reality software such as modeling for construction planning, urban planning, civil engineering and traffic modeling (Figure 1).

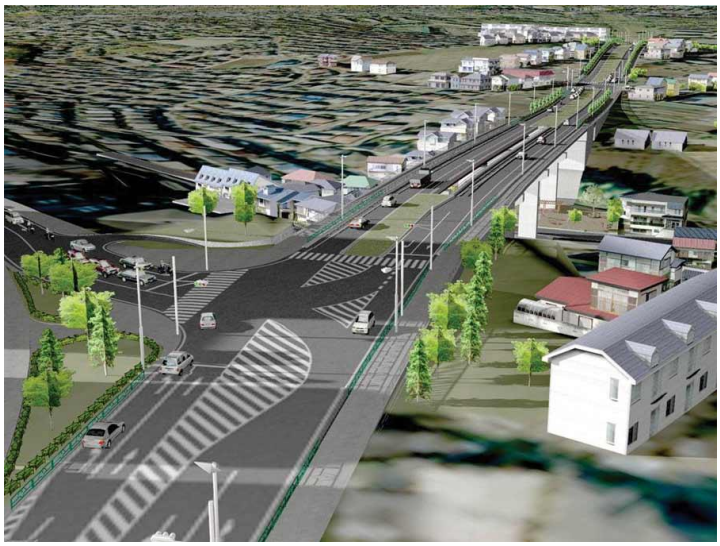


Figure 1: UC-win/Road

So far, the interaction between the Real World and the Virtual World was made with common input devices like keyboard , mouse, joystick or more recently 3D mouse. To offer a more natural interaction with the Virtual World, we developed an interface for UC-win/Road that allows the user to directly interact at different levels with the Virtual World, ranging for a simple interaction trough a monitor to a full immersion through a Head Mount Display (HMD). At the highest level, the user is immerged in the Virtual World, but contrarily to existing applications based on characters to represent the user, a realistic representation of his own body will be available as visual feedback.
This kind of interface was made possible due to the recent advances in sensing, particularly when Microsoft released

the Xbox video game console using the Kinect, a 3D depth sensor to detect the user's motion (Figure 2). The main concept behind this new generation of video game consoles is to turn the user's body into an input device, allowing the control of the video game only by body movements and hand gestures. Our developments started with the Kinect sensor, and later were followed with the Xtion Pro depth sensor (Figure 2). It uses the same technology than the Kinect, but is smaller, lighter, and USB powered, which makes it more fit for small robotics applications.

Figure 2: Devices supported by the Kinect Plugin:

Kinect, Xtion Pro, Xtion Pro Live (Xtion Pro + RGB + Audio)

The main feature of these sensors is the ability to create a 3D mapping of the environment. It uses the structured light imaging principle: an infrared laser (frequency slightly below red light) continuously projects a predefined dot pattern while a camera simultaneously records the deformation of the pattern when an object is in the sensor's field of view. The deformation is then used to generate a 3D map of the environment.

The figure 3 shows the output of the RGB camera (Figure 3.a), the depth map (Figure 3.b) and the combined outputs (Figure 3.c).

Figure 3.a: RGB Camera Output

Figure 3.b: Depth Map

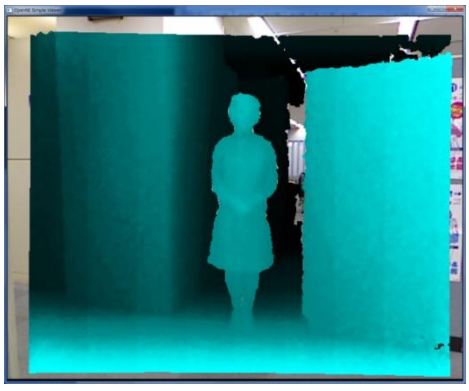


Figure 3.c: RGB Output + Depth Map

After a description of the Kinect Plugin capabilities, we will present the applications of the plugin in the context of Robotics: tele-operation of a robotic arm and tele-operation of a small RC car, before concluding the paper.

2. Kinect Plugin Capabilities

2.1. Overview

The fundamental data provided by the Kinect (Xtion) sensor is the raw depth map data, a 3D representation of the environmenent seen by the sensor. The depth map is then analyzed to provide more conplex data. An important feature is the ability of the sensor to identify human body shapes and seprate thme from the background. A time analysis of this data provide more advanced capabilities, like the motion capture of users, by the mean of 3D skeleton tracking.
The table 1 shows the functionalities available in the OpenNI SDK and the ones provided through the Kinect Plugin:

Function	Open NI	Kinect plugin
Depth map	●	●
User data	●	●
Skeleton tracking	●	●
Air driving	×	●
Gesture interface	×	●
Virtual grabbing	×	●

Table 1: Functionalities available in OpenNI and through the Kinect Plugin

The main advantage of the Kinect Plugin is the ability to access the all raw Kinect data and more advanced data in a very simple way, directly by a UDP connection, thus removing the burden of the implementation of the data access from the sensor and processing. For each capability, a specified data server is accessible, providing directly the required data.
All the available data (raw and processed) of the Kinect Plugin are displayed in a main window with a tab window for each data (Figure 4.a)

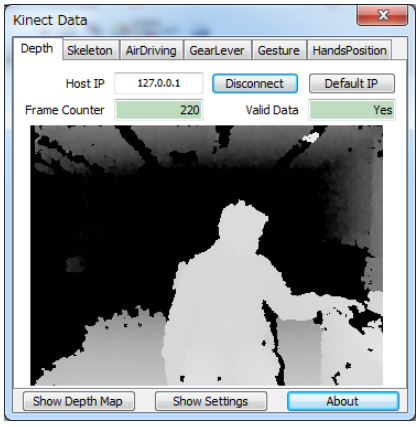


Figure 4.a: Main Plugin Window

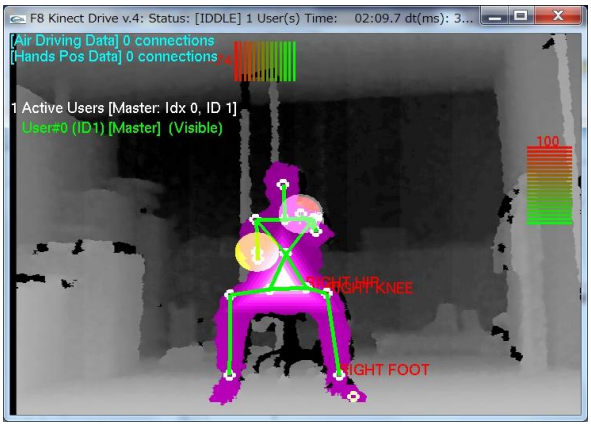


Figure 4.b: Depth Map Window

In addition, an auxiliary window shows the processed data merged together for a specific application (Depth Map window), for instance the data for the Air Driving application overimposed on the depth map data (Figure 4.b)

2.2. Raw Depth Map Data

The basic data from the sensor is the raw depth map data. The output is a "depth image" of the environment in front of the sensor, with a VGA resolution. The value at each pixel (X, Y) is the distance from the sensor to the closest object at the coordinates (X, Y), in mm. The Figure 5 shows the raw depth map data in linear gradient color: the closer objects are to the sensor, the brighter they appear.
The value of any of the points of the depth map can be checked simply by using the mouse (Figure 5); the value of the depth map d(x, y) at the position (x, y) is displayed near the position of the mouse cursor:

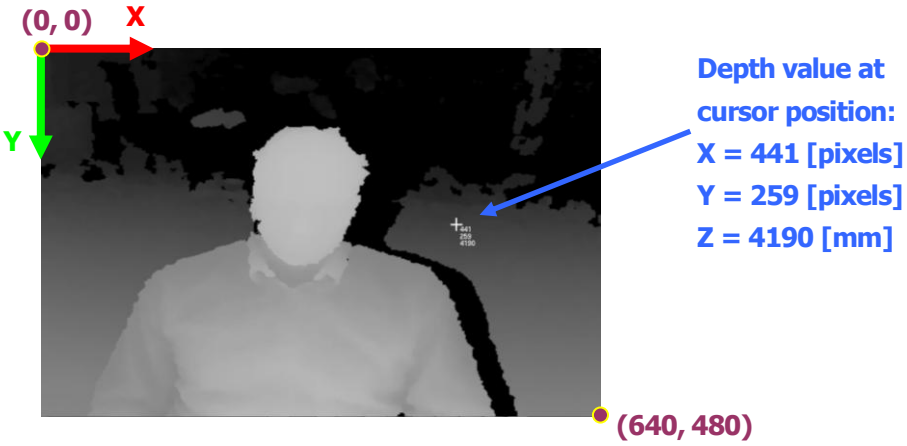


Figure 5: Raw Depth Map Data (User not detected yet)

2.3. User detection

From the previous raw depth map data (Figure 6), it is possible to identify and separate the shape of a human body from the background. When the program is started, the user needs to move in order to be detected as a human user. Once the user is detected, his position can be estimated, approximately at the "center of gravity" of the area of the detected user.

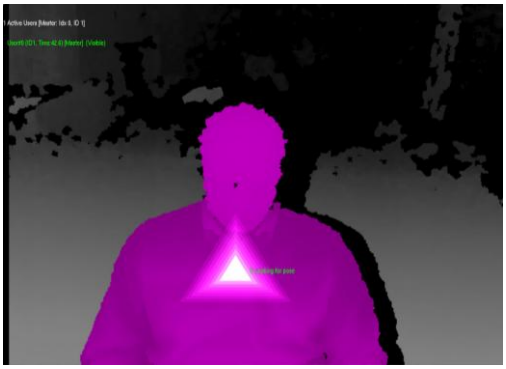


Figure 6: User detection (in pink) and master user (white triangle)

Each user is then displayed with a different color (Figure 6), and is given an individual identifier (User ID). To avoid conflicts in the user interface operation, only one user can operate the interface at a given time. The user controlling the interface becomes the master user, labeled with a white triangle. The user who is the closest to the sensor becomes automatically the master user.

2.4. Skeleton Data

One of the main functions of the Kinect sensor is the ability to track in 3D the joints of the human body (skeleton tracking). The Figure 7 shows the tracking of the user's body joints, with the resulting skeleton model shown as a simplified model of the user's skeleton.

The tracking of the skeleton joints is possible when the body joints above the hip joints are visible by the sensor (hip joints included). A practical case is when the user is sitting at a desk and the sensor is positioned on the desk, in front of the user.

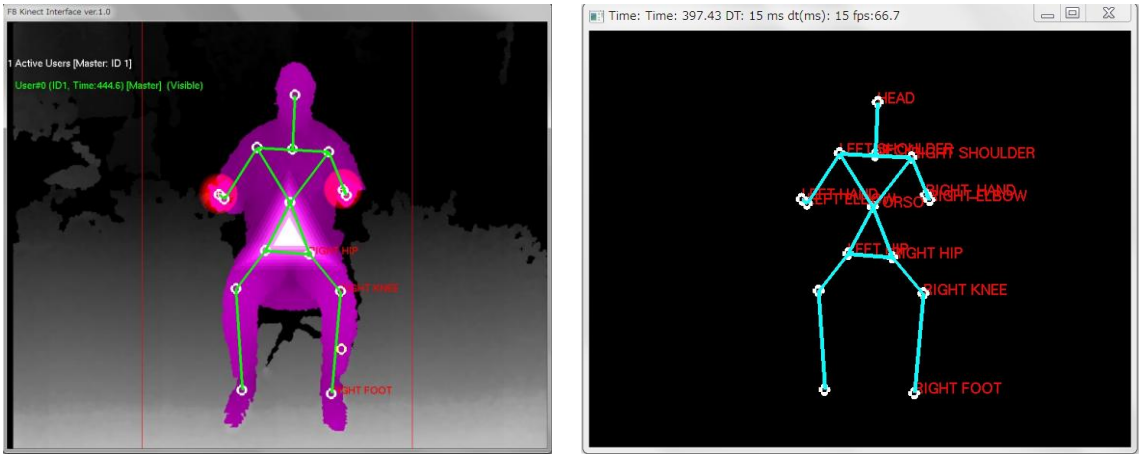


Figure 7: Skeleton tracking on the server application (left) and client application (right)

The skeleton data can be examined in real time in detail by the user of the Kinect Plugin window (Figure 8):

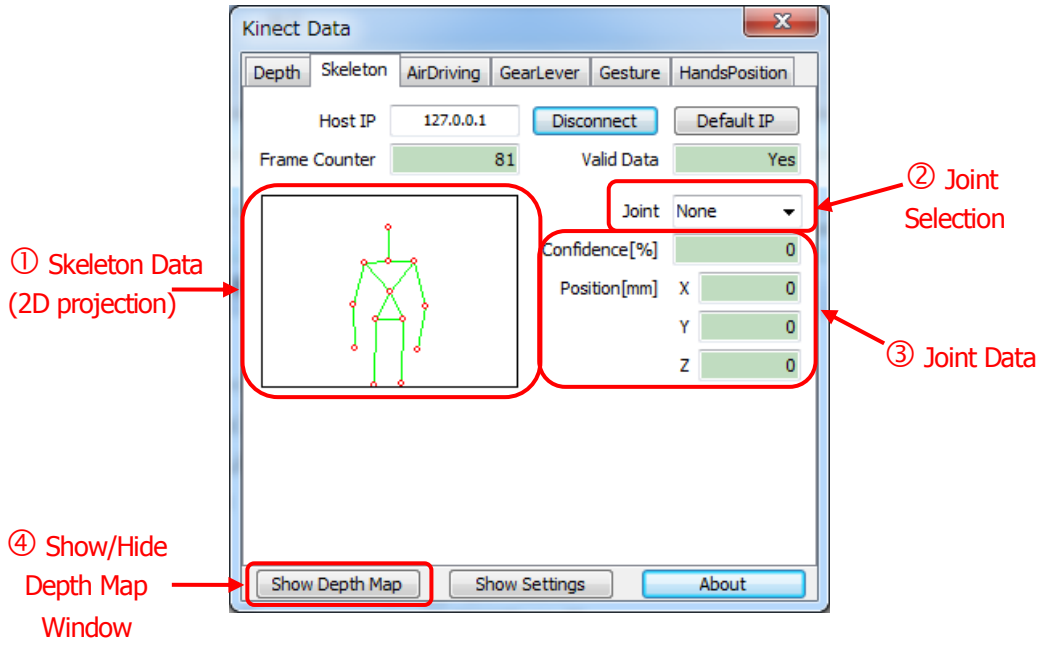


Figure 8: "Skeleton" data

The possible applications are 3D motion tracking, robot tele-operation and Human-Machine Interface.

2.5. Air Driving Data

The air driving concept is simple: drive a vehicle without the use of any input device. By tracking the hands and the right foot of a sitting (or standing) user, it is possible to estimate a steering wheel angle and an acceleration/braking factor corresponding to the pressure of the foot on the accelerator (or brake) pedal (Figure 9).

Once the user is detected and the skeleton tracking has started, the user controls the vehicle the same way than a real car/truck, except that no steering wheel or accelerator/brake pedal is used. Instead, the drivers move his hands just as he was holding a steering wheel and presses or releases the break/gas pedal. When the detection is accurate enough (best when seated), the user can smoothly control the gas pedal and brake pedal.

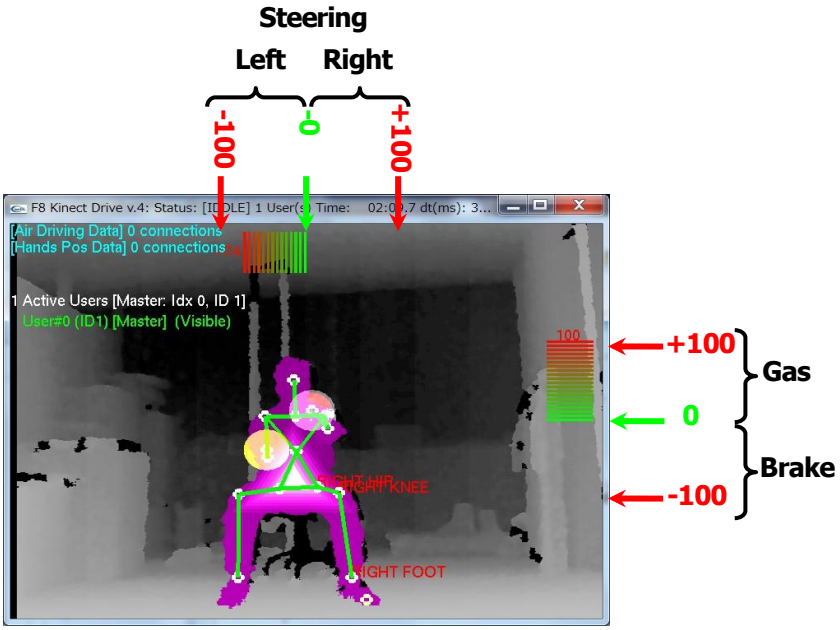


Figure 9: Air Driving operation

The steering angle is displayed at the top of the depth map window. The steering factor is within [-100; 0] when turning left and [0; +100] when turning right (Figure 9). The pressure on the gas pedal is represented on the right side of the window. The accelerator factor is within [-100; 0] when braking and [0; +100] when accelerating.

The application of the Air Driving is the driving of a vehicle in a simulated environment (Figure 10) and Robot Tele-operation (Figure 15).

Figure 10: Air Driving at Tokyo Game Show Expo (2012)

The value of the Air Driving data can be checked from the Kinect Plugin window (Figure 11):

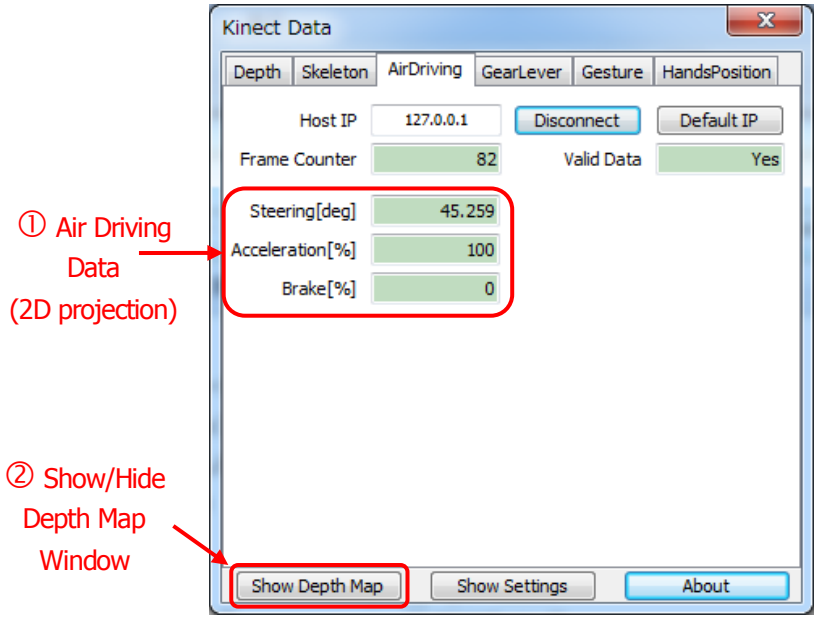


Figure 11: "AirDriving" data

2.6. Virtual Gear Lever

The purpose is the detection of the action of grabbing of a virtual object in a simulation. The current application is the gear change (Forward/Backward) in the Air Driving application (Figure 9): a virtual gear lever allows the user to change gear from forward to backward, by a movement identical to grabbing a gear lever and pushing it forward (forward gear) or backwards (backwards gear).

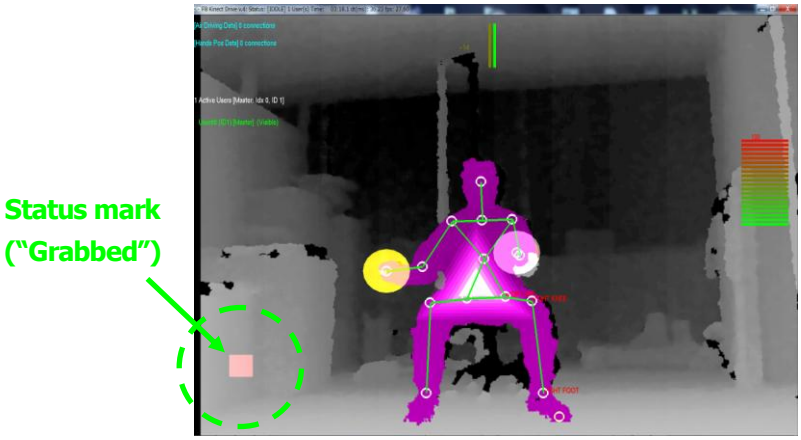


Figure 12: Air Driving. The user grabs the virtual gear lever (square-shaped indicator in the bottom left side of the window)

When the user simulates the action of grabbing a virtual gear lever, a red square appears at the bottom left side of the window to notify the user that the action was detected (Figure 12). The virtual gear lever status is changed to "GRABBING" status. Until that, the default status is "STEERING". If no action is detected from the grabbing position within 2 seconds ("Persistence Time"), the action is cancelled and the status is restored to "STEERING".

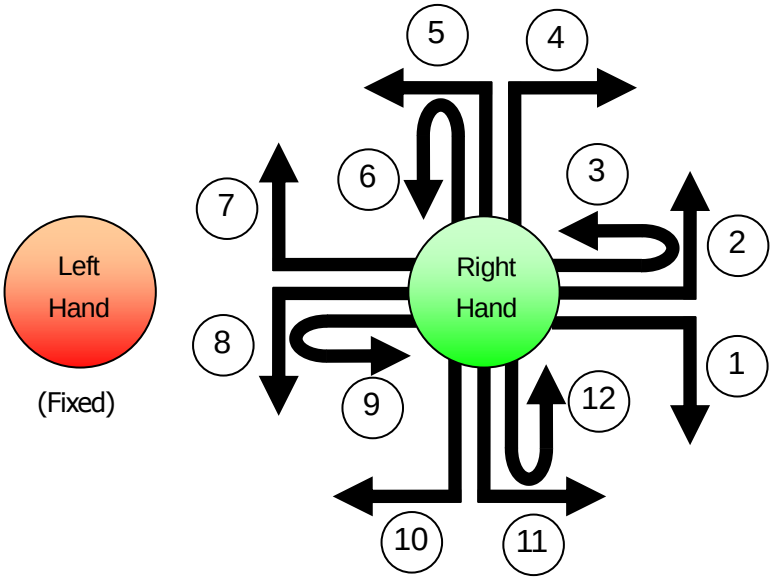
From the "GRABBING" status, the user needs to move his hand forward (resp. backward) to change the gear to forward gear (resp. backward gear). If successful, the gear lever status becomes "FORWARD", resp. "BACKWARD" (Figure 13).

More generally, the virtual gear lever function can detect the grab and release actions, so it can be used to manipulate objects in the virtual environment.

Figure 13: Air Driving. The user pushes the virtual gear lever forward

2.7. Gesture Interface

The gesture interface allows a simple Man-Machine interaction by using elementary hand movements performed with the right hand. An elementary movement is defined as a straight line (UP, DOWN, LEFT or RIGHT), while a gesture is defined as a combination of two elementary movements, performed continuously with the right hand, at the same pace. The available combinations are shown on the Figure 14.



1	RIGHT/DOWN
2	RIGHT/UP
3	RIGHT/LEFT
4	UP/RIGHT
5	UP/ LEFT
6	UP/ DOWN
7	LEFT/ UP
8	LEFT/ DOWN
9	LEFT/ RIGHT
10	DOWN/ LEFT
11	DOWN/ RIGHT
12	DOWN/UP

Figure 14: Identified gestures of the Gesture Interface

3. Applications in Robotics

3.1. Overview

The Kinect/Xtion Pro sensors provide accurate 3D depth data at a very low price regarding previous techonologies (laser, stereo camera). This explains the interest for this sensor from many developpers, who used the sensor for many types of applications, ranging from motion capture to synchronous localization and mapping (SLAM)[1]. Our first developments were oriented towards the simulation and control of small robots: a robotic car and more recently a robotic arm. The purpose is to remotely operate those robots in a very natural way, allowing for instance to send them in a hazardous environment while keeping the user safe in a remote environement.

3.2. Robotic Car

Recent progress in the automotive field have brought to the market cars equipped with sensors used until now for robotic purposes, mainly for safety concern. Cars got equipped with distance sensors for automatic braking system, night vision cameras for pedestrian detection and so on. The most famous example is the Google driverless car, which is a fully autonomous car, currently allowed to be used legally in four different states of the U.S. (in January 2014)[2]. In this context we used the RoboCar, a model car based on a RC car, with added sensors (stereo camera, IMU, laser range finder). To demonstrate the tele-operation of a car-like vehicle, we simulated the RoboCar in a virtual environment, while the actual car evolves on the ground (Figure 15).

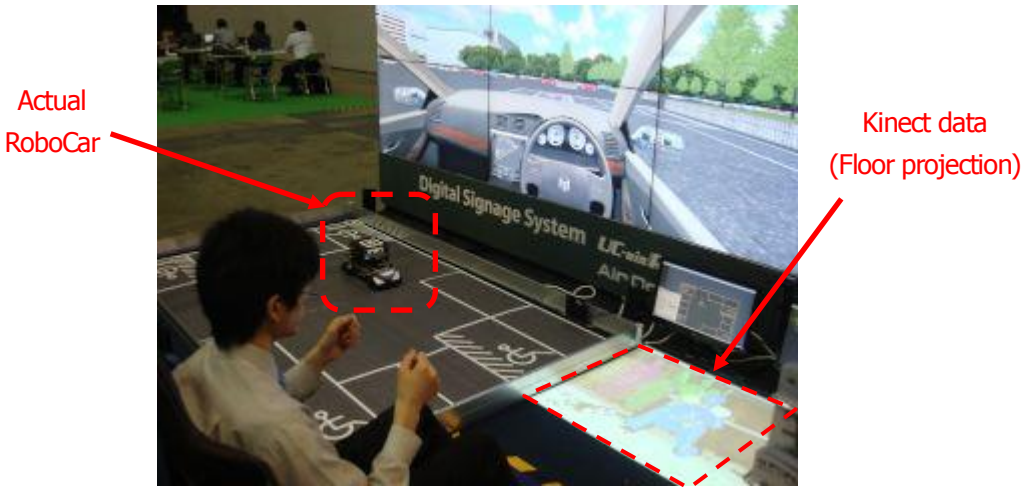


Figure 15: Air Driving at Robotech Expo (2012)

The RoboCar proved to be useful for our developments but with few limitations, like its weight or a communication over the Wi-Fi prone to discontinuities when evolving in a noisy Wi-Fi environment.

To solve those issues, we started to develop a much lighter car, the "Lily Car" (contraction of "Liliput Car"). The car will have a modular structure, i.e. the car will have a main body, with interchangeable modules depending on the application. The figure 16 shows the Lily Robotic car at the prototyping phase.

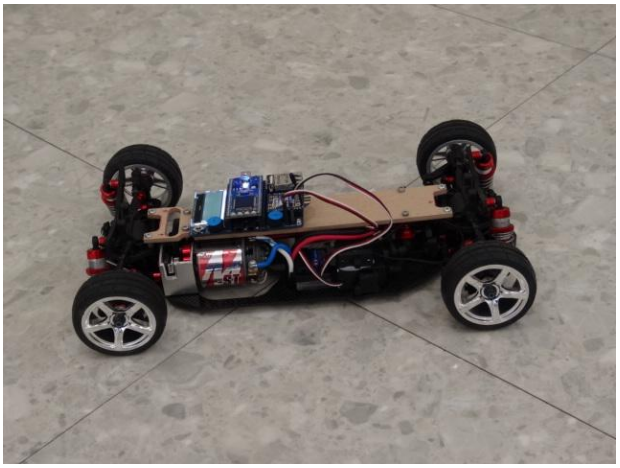


Figure 16: Lily Robotic Car

The car is equipped with the following sensors:

- incremental encoders to estimate the rotation speed of driving wheels
- a 2.4GHz transceiver for wireless communication
- an accelerometer
- a temperature sensor to detect overheat situations
- a camera with a 5.8GHz transmitter for live feed

- an LCD and mini joystick for manual settings directly on the car

With this car we hope to build a development platform (simulation in VR and actual car) for autonomous functionalities for smart cars, for instance the automatic parking or assistive braking system.

3.3. Robotic Arm

Another that we are currently investigating is the tele-operation of a robotic arm based on the skeleton joint data from the Kinect Plugin. Such development could have potential application in various sub fields of Robotics, ranging from tele-surgery to tele-operation of manipulators in nuclear plants.

In our development we used a 5DOF robotic arm (AL5D from LynxMotion), shown in Figure 17. For demonstration purposes, we try to apply the skeleton tracking data to the control of the robotic arm.

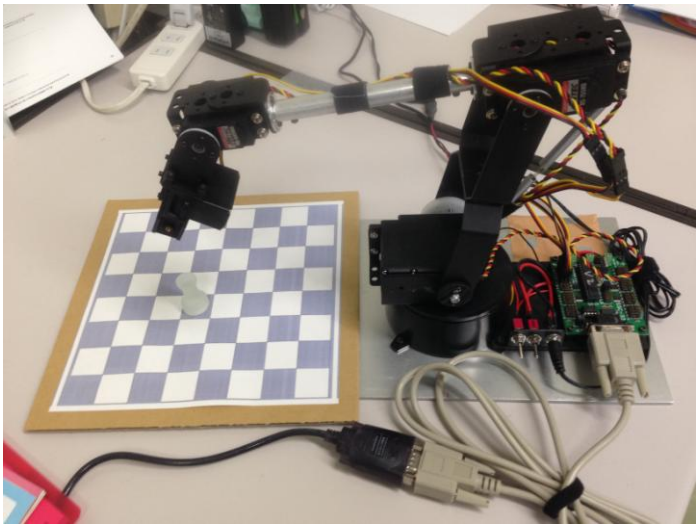


Figure 17: AL5D Robotic Arm

In this application, a simple simulation was made of the operation of the AL5D (Figure 18), mainly for safety reasons for the user and to avoid mechanical damage to the arm. We retrieved the skeleton data directly from the Kinect Plugin by UDP connection.

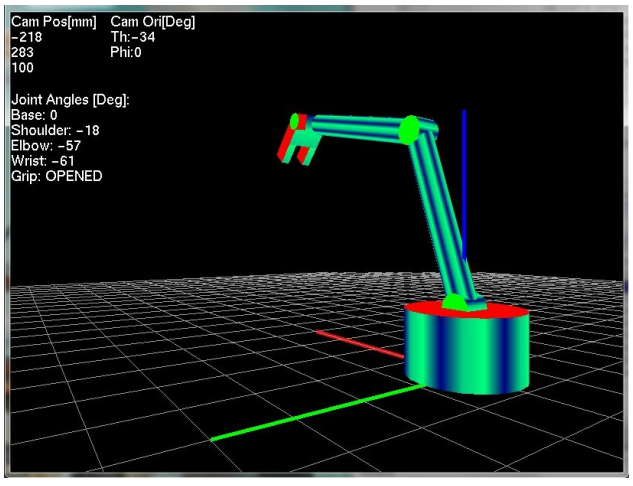


Figure 18: Robotic Arm simulation (client application)

The joints data included in the skeleton data are then mapped into the robotic arm space, to estimate its state vector (angle of each joint and opening of the grip).

The Figure 19 shows the Kinect Plugin window on the right side (UDP server), and the client application using the skeleton data

to control the robotic arm (on left side).

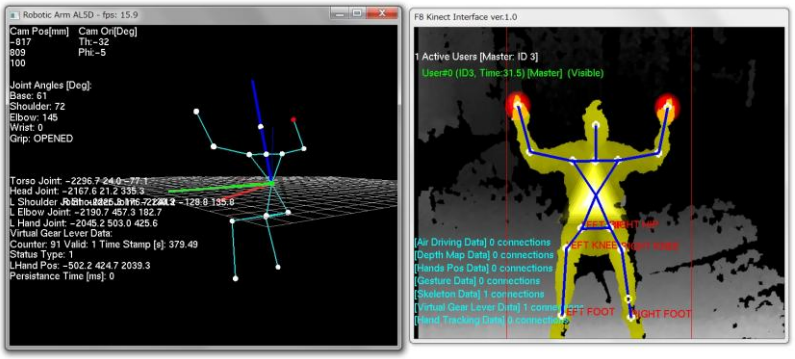


Figure 19: Client application (left) and Kinect Plugin server (right)

By using the virtual gear lever function of the Kinect Plugin it is possible to detect the grabbing action performed by the user, which controls in turn the opening/closing of the grip of the robotic arm, allowing the robotic arm to catch objects. As a result the robotic arm can be fully controlled by the user, without using any input device. At the current stage of development, the control was assessed in simulation. Moreover, we started tests on the actual robotic arm. It can be controlled remotely by the user and we are currently improving the control algorithm, to have a smoother control of the robotic arm.

5. Conclusion

In this paper we presented the Kinect Plugin, and interface for the UC-win/Road VR software, and two applications in the robotics field. The data from the plugin allowed to control a small smart car and a robotic arm. We hope in the future to pursue the efforts in applications involving Robotics and Virtual Reality and on a greater extent, man-machine interface applications.

References

[1] A solution to the simultaneous localization and map building (SLAM) problem, IEEE Robotics and Automation, Vol. 17:3

[2] T. Litman, Autonomous Vehicle Implementation Predictions Implications for Transport Planning**,** http://www.vtpi.org/avip.pdf