Parrot

Compiling the A.R. Drone SDK demonstration program on Microsoft Windows

1. Prerequisites

This demo was tested with Microsoft Visual C++ 2008 Express edition, on a Windows 7 64Bits Platform.

It should work with any version of Windows XP, Vista, and Seven with minor changes if any.

The required libraries are:

- The Microsoft Windows SDK (available here), "Windows Headers and Library".
- The Microsoft DirectX SDK (available here), to use a Gamepad for drone control.
- The SDL Library (available here) which is used in the example to display the video, but which can be easily replaced;
- [Only for Windows XP and earlier] The Pthread for Win32 library (available here)

2. Getting development files

You should have downloaded and unpacked the following elements :

- the ARDroneLib directory (with the Soft, VLIB and VP_SDK subdirectories)
- the Examples directory and specifically its Win32 subdirectory, which contains:
  - the demonstration source code in sdk_demo
  - the Visual C++ 2008 Express solution in VCProjects

3. Setting up the development environment

a. Making the project aware of the code location on your computer

- Open the Visual Studio solution (file *\Examples\Win32\VCProjects\ARDrone\ARDrone.sln*).
- Open the Property Manager tab (next to the Solution Explorer and the Class View tabs).
- Double click on any of the ArDrone_properties entry to edit it.
- Go to Common Properties -> User Macros
- Edit the **ARDroneLibDir** macro so its contains the path to the ARDroneLib directory on your computer.
- Edit the **Win32ClientDir** macro so its contains the path to the demonstration source code directory on your computer.

Note : you can also directly modify those paths by editing the ArDrone_properties.vsprops file with any text editor.

b. Making the project aware of the libraries location on your computer

- In Visual Studio up to version 2008, the Tools->Options->Environment->Projects and Solutions->VC++ Directories must contains the paths to the Include and Libraries files for the above mentioned prerequisite libraries.
- In Visual Studio 2010, these directories must be set in the Project settings.

4. Required settings in the source code before compiling

a. Operating System related settings

In the Solution Explorer, search for the *vp_os_signal_dep.h* in the ARDroneLib project.

In this header file, one of the two following macros must be defined:

- Use `#define USE_WINDOWS_CONDITION_VARIABLES` to use Microsoft Windows SDK for thread synchronisation (you must have Windows Vista or later)
- Use `#define USE_PTHREAD_FOR_WIN32` to use pthreads for synchronisation (mandatory if you have Windows XP or earlier, possible with later Windows versions)

b. Drone related settings

The *win32_custom.h* file contains the IP address to connect to. It's 192.168.1.1 by default, but the drone actual IP address might be different if several drones use the same Wifi network (drones then pick random addresses to avoid IP address conflicts).
Check your drone IP address with pings or telnet commands before running the example.

c. Gamepad related settings

The *gamepad.cpp* file contains code to pick the first gamepad that can be found on the computer and which is supported by the DirectX subsystem. In the example, buttons are statically linked to drone actions for three specific gamepads. Other gamepads have no effect.
Please modify this code to enable the drone to move with your own input device (code should be self-explanatory).

5. Compiling the example

Simply generate the solution once the above mentioned settings were done.

6. What to expect when running the example

Before running the example, one of the computer network connections must be connected with the drone. Pinging the drone or connecting to its telnet port MUST work properly. The drone MUST NOT be paired with an iPhone (using the drone with an iPhone prevents the drone from accepting connections from any other device , including a PC, unless the *unpair* button (below the drone) is pressed (which is acknowledged by the drone by making the motor leds blink).

If the network is correctly set, running the example will display a console window showing which commands are being sent to the drone, and a graphic window showing the drone broadcasted video.

The example is basic one which simply hangs by saying "Connection timeout" if no connection is possible. Check the network connectivity and restart the application if such thing happens. The video window does not respond if no video is received from the drone. Close the console window to close the application, or deport the SDL window management code from the video processing thread to an independent thread.

7. Quick summary of problems solving

Q. The application starts, but no video is displayed, or “Connection timeout” appears.
A. The client was to slow connecting to the drone, or could not connect. Restart the application, or check your network configuration if the problem remains.

Q. I can’t open any source file from the solution explorer
A. Make sure the **ARDroneLibDir** and **Win32ClientDir** macros are set in the Properties Manager

Q. I get the “Windows.h : no such file or directory” message when compiling.
A. Make sure the Windows SDK was correctly installed. A light version should be installed along with VC Express.

Q. I get the “Cannot open input file ‘dxguid.lib’ message when compiling.
A. Make sure the DirectX SDK was correctly installed

Q. I get the “Error spawning mt.exe” message when compiling.
A. There is a problem with the Windows SDK installation. Please reinstall.