Sixense SDK Overview

Sixense Control System Runtime Library

Table of Contents

Library Overview

Description

sixense.dll is the software driver that allows PC applications to communicate with the Sixense Control System.

Files

The following files are related to sixense.dll.

Filename	Description
SixenseSDK/include/sixense.h	Main sixense.dll header file.
SixenseSDK/lib/win32	Windows API libraries.
SixenseSDK/samples/sixense_simple3d	WIN32 sample application.
SixenseSDK/doc	Sixense documentation.
SixenseSDK/src/sixense_simple3d	Source code for simple3d sample application

Sample Programs

SixenseSDK/sample/sixense_simple3d
SixenseSDK/src /sixense_simple3d

A simple test application based on freeglut. Source code is provided as a reference on how to use the Sixense core API as well as the Sixense Utility Library.

Coordinate System

Coordinate System for Positions

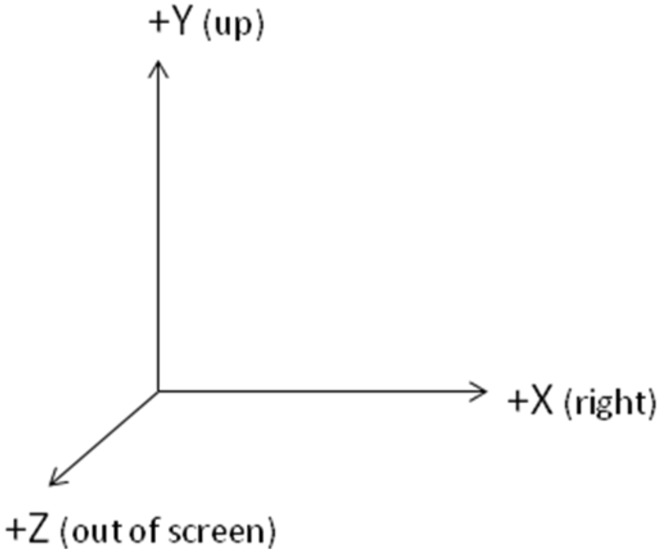


Figure 1: Sixense coordinate system

Rotations

The sixenseControllerData structure contains a 3x3 rotation matrix for each controller This matrix is stored in column-major order, so the 3 values of the first column are `mat[0][0], mat[0][1] and mat[0][2].`

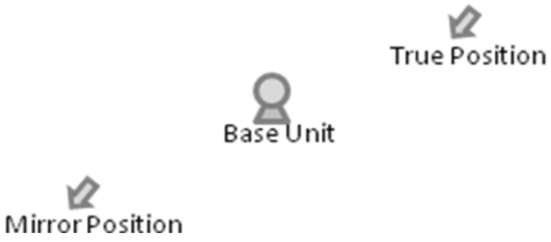
The three columns of the rotation matrix represent the three transformed unit axes, so column 0 (mat[0][]) is the rotated X axis, column 1 (mat[1][]) is the rotated Y axis and the third column (mat[2][]) is the rotated Z axis. If the matrix were a 4x4, the positions would be stored in the fourth column (mat[3][]).

When the controller is held flat with the floor, with the buttons aiming at the ceiling and the controller aiming at the TV (and with the base unit's USB cable aiming at the TV), mat will contain the identity matrix, $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$. Aiming the controller at the ceiling will result in mat being $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$, meaning the X axis remains the same, the Y axis rotates to point out of the screen, and the Z axis rotates down to point at the floor.

Hemisphere Tracking

Overview

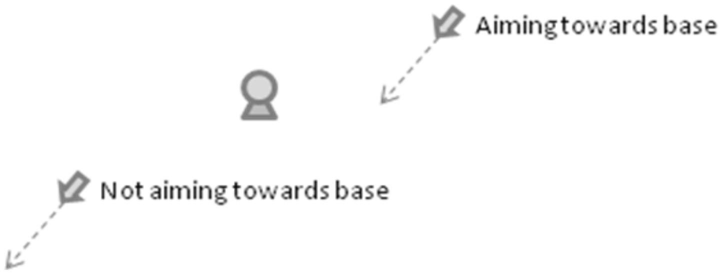
Due to ambiguities in the position solution, Sixense technology reports two positions each frame: the true position of the controller, as well as the position mirrored directly across the base unit. The orientation is the same for both positions.



When first powered on, the system does not know which of the two is the true controller position. This ambiguity is resolved by one of two techniques.

Hemisphere Tracking

The most common is known as Hemisphere Tracking. The first step is to determine which of the two positions is the correct one. For this, user input is required. The player is told to 'aim the controller at the base unit' and press a button. When the button is pressed, only one of the two position solutions is pointing towards the base so it can be selected as the true solution.



From frame to frame then on, Hemisphere Tracking relies on the fact that the controller is unlikely to have moved very far, so the position selected in any given frame is the one that is closest to the position in the previous frame. This assumes that the system is actively tracking, if the system is turned off for a brief time it can no longer be assumed that the controller didn't move to the far side of the base and the user must be prompted again. When hemisphere tracking is enabled, the `sixenseControllerData` field `hemi_tracking_enabled` will be set to 1.

Enabling Hemisphere Tracking

Hemisphere tracking is disabled by default. When the system is powered on the controller positions will ‘pop’ if the controllers are moved below the height of the base unit. The system should not be left in this state. Instead, one of these techniques should be used.

Controllers in Dock

For systems that have a docking station like the Razer Hydra, Hemisphere Tracking is enabled automatically when the controller is placed in the dock. Placing a controller in the dock will also designate the controller as ‘left’ or ‘right’.

Controller Manager

The controller manager portion of the sixenseUtils DLL can be used to manage the process of instructing the player to sequentially aim the left and right controllers at the base and pull each trigger, setting both hemisphere tracking to enabled and designating left and right controllers. See the source code for the sixense_simple_3d application for an example of using the controller_manager.