ARToolKit による紙ピアノの実装

K06 立松(pine)

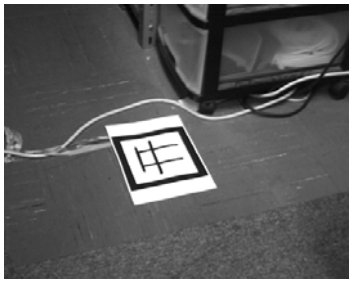
1.はじめに

皆さんは、ARToolKit って知っていますか？
ARToolKit とは、AR(拡張現実：現実世界に仮想世界を重ね合わせる技術)を実現する為の、C 言語のライブラリです。･･･といっても、ピンとこないですね。詳しい紹介のあるサイトをご紹介します。

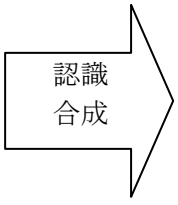
工学ナビ　ARToolKit を使った拡張現実感プログラミング

http://www1.bbiq.jp/kougaku/ARToolKit.html

そう。ARToolKit は、Web カメラでマーカを認識すると、ディスプレイに実際の世界と 3D オブジェクトを重ね合わせて表示するソフトを簡単に作れるライブラリです。マーカを動かすと、実際に３D オブジェクトもぐりぐり動きます。ニコニコ動画や、YouTube などで、「ARToolKit」と入力してみるのもいいかもしれません。実際のモノが見えますよ。



実際の世界
(四角いのがマーカ)

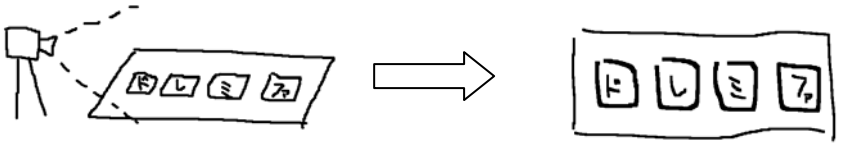


３D オブジェクトが合成された世界
円筒状の物体がオーバーレイされている

2.紙でピアノ

この ARToolKit を使って、ピアノを作ってみようと思います。
今回は、**3D オブジェクトの重ね合わせは利用しません。**

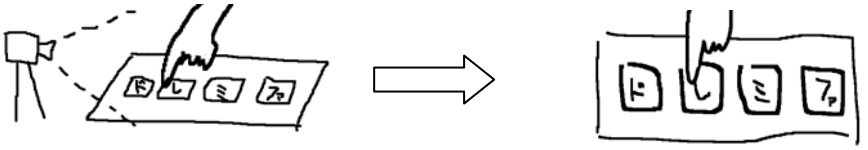
マーカ認識機能だけを用います。イメージとしては、



紙（マーカ）をカメラで写す

カメラではこんな感じに認識

ド・レ・ミ・ファのマーカは隠されていない　→　**何もしない**



レのマーカは隠されている　→　**レを発音する！**

こんな感じのものを作っていきます。

3.準備しましょう

ARToolKit を用いて開発をする場合、まず ARToolKit(および glut)を開発環境下にセッティングしなければいけません。

ダウンロードや、インストールは、前述した**工学ナビ**さんの Web を参考にして行うといいでしょう。

しかし、一つ問題があります。工学ナビさんでは、VisualC++6.0 環境で、コンパイラの設定をしています。

昨今の多くの方は VisualStudio2005 を使用されていると思いますから、当然設定は違います。注意してくださいネ。

4.ARToolKit を用いた開発

とくに難しいことはしないので、ARToolKit についてきたサンプルソースを改造して、紙ピアノを製作することにします。まず、simpleTest.c を開きましょう。C 言語の、見慣れた main 文が目に入りますね。工学ナビさんの web を参考にして、解読していきましょう･･･

あ、今回は、ARToolKit の花とも言える、３D の描画関数 draw()は使用しません。ズバーッと消しちゃいましょうｗ

気を取り直して、頭のほうからずらっと読んでいきます。
マーカのパターンは、グローバル変数で宣言され、読み込まれているようです。

```
char           *patt_name      = "Data/patt.hiro";
int             patt_id;
double          patt_width     = 80.0;
double          patt_center[2] = {0.0, 0.0};
double          patt_trans[3][4];
```

こいつを増やせば、複数のマーカのパターンを扱うことができますね。
あまりスマートではないですが、

```
//patt a
char           *patt_name_a      = "Data/do.patt";
int             patt_id;
double          patt_width     = 80.0;
double          patt_center[2] = {0.0, 0.0};
double          patt_trans[3][4];
//patt b
char           *patt_name_b      = "Data/re.patt";
int             patt_id_b;
double          patt_width_b     = 80.0;
double          patt_center_b[2] = {0.0, 0.0};
double          patt_trans_b[3][4];
```

こういった感じで、増やせそうです。
関数 init()の中にも、対応した箇所があります。増やしていきましょう。
更に、関数 mainLoop()の中で、マーカの判定を行っていますね。マーカそのものと、判定の箇所も、上記の要領でどんどん増やしていきましょう。

っと、そんな要領で必要な箇所はすべて増やしましたか？
（初期化の部分も忘れてはいけません。）
お次は、音を鳴らす部分について説明します。
今回、音を鳴らす方法に、MIDI を用います。具体的に言えば、Windows がもっている「ソフトウェアシンセ」というやつですね。

```
#include <mmsystem.h>
#pragma comment(lib, "winmm.lib")
```

MIDI を使うには、この「おまじない」をソースファイルの先頭に追記しておきましょう。

そして、MIDI デバイスのオープン処理を行います。midi_init()とか、テキトーな名前の関数を作って、main 文の最初のほうで実行しておきましょう。MIDI デバイスの操作については、以下の Web が参考になります。

Windows プログラミング研究室

http://lcl.web5.jp/prog/midimsg.html

発音はこんな感じです。これも関数にまとめて、マーカ判定の場所から読み出しましょう。

```
static void midi_out(int flg){
          switch (flg) {
          case 0:
                    midiOutShortMsg(g_hMidi, 0x00403c90);
                    //(3c)16は(60)10  つまりド。
                    break;
          case 1:
                    midiOutShortMsg(g_hMidi, 0x01403e90);
                    //(3e)16は(62)10  つまりレ。
                    break;
```

消音に関しても同じです。

```
static void midi_stop(int flg){
          switch (flg) {
          case 0:
                    midiOutShortMsg(g_hMidi, 0xB07800c0);
                    //ド off。
                    break;
          case 1:
                    midiOutShortMsg(g_hMidi, 0xB17800c0);
          //レ off
                    break;
```

このソースでは、ドは 0 番ポート、レは 1 番ポートに出力をしています。0x00 とか 0xB0、0x01 とか 0xB1 のところですね。

発音や消音は、フラグ変数を作ってちゃんと管理しましょう。

最後には、MIDI の close 処理を行います。
細部を煮詰めて、あとは楽しいデバッグのお時間です。
ちなみにわたしは、デバッグで 1 日費やしました。
皆さんは、ちゃんときれいなコードを書きましょうｗ

あと､気をつけないといけないのはコンパイル時のリンクの設定です。ARToolKit は､依存している lib ファイルや DLL ファイルが多くあるので、サンプルプログラムの設定とにらめっこしてコンパイルを行ってください。

開発はザッとこんな感じです。わたしのコーディングの汚さと、紙面の都合で、カナ～リ端折りましたが、だいたいこんな感じで組んでいます。

5.まとめ

このように､ARToolKit は C 言語の知識だけでホイホイと書けちゃいます。C++や、Windows プログラミングに深い知識がない人でも、アイディア次第で様々な AR アプリケーションの作成が可能です。

この紙ピアノを試してみたい人向けに、バイナリを公開しています。
Web カメラ(USB カメラ)を用意し、次をダウンロードしましょう。
http://aitech.ac.jp/~k06061w/kami_piano.zip

ファイルを解凍したら、kami_piano.exe を実行します。
楽譜のファイル（kenban.jpg）を印刷して使用してください。

実際の様子を、ニコニコ動画にもアップロードしてあります。

http://www.nicovideo.jp/watch/sm2400851

以上です。お粗末様でした。またお会いしましょう！