NtKinect: C++ Class Library for Kinect V2

新田 善久[1,a)]

概要：Kinect for Windows V2 は骨格認識，顔認識，音声方向取得など多くの機能を持ったデバイスである．そのプログラミング開発を容易にするために C++ 用のクラスライブラリ NtKinect を開発し，オープンソース化した．NtKinect を用いたプログラムはマルチスレッド対応，DLL 化可能で，他の言語や開発環境からも Kinect V2 の機能が容易に利用できる．その設計方針について論じる．

キーワード：Kinect V2, C++, DLL, Unity

NtKinect: C++ Class Library for Kinect V2

Nitta Yoshihisa[1,a)]

Abstract: Kinect for Windows V2 is a device with many functions such as skeleton recognition, face recognition and voice direction acquisition. In order to make its programming easier, we developed NtKinect class library for C++ and made it open source. Programs using NtKinect are multi-threaded, DLLable, and Kinect V2 functions can be used from other programming languages and environments. We will discuss its design policy.

Keywords: Kinect V2, C++, DLL, Unity

1. はじめに

Microsoft が開発した Kinect for Windows V2 [*1] [1] は骨格認識，顔認識，音声方向取得など多くの機能を持ったデバイスである．この Kinect V2 を利用したプログラミングを容易にするために，C++ 用のクラスライブラリを開発し，オープンソースとして公開した．その設計方針と API について論じる．

2. Kinect for Windows V2

Microsoft は Kinect V2 のプログラミング開発用に Kinect for Windows version 2.0 SDK を配布している．その C++ 用のリファレンス [2] の native code API にある *Interface* [*2]の一覧を表 1 に示す．*Interface* 名は機能を表す接頭子で始まり，大きく 9 種類に分けられる．*Interface* は全部で 54 種類あり，その中に含まれているメソッドの総数は 277 である．

表 1 Kinect SDK の *Inteface*

接頭子	機能	種類	総メソッド数
IAudio	音声	10	50
IBody	骨格認識	11	61
IColor	色カメラ	6	31
ICoordinateMapper	座標変換	2	17
IDepth	深度センサ	5	26
IInfrared	赤外線センサ	5	22
IKinectSensor	デバイス本体	2	19
IMultiSource	複数ストリーム	4	16
その他	その他	9	35
合計		54	277

この SDK を直接用いたプログラム開発 [3] は，自由度が

[1] 津田塾大学
Tsuda College, 2–1–1, Tsuda, Koadira, Tokyo 187–8577, Japan
[a)] nitta@tsuda.ac.jp

[*1] 以下 Kinect V2 と記す．

[*2] ここでは C++におけるクラスと同義．

表 2　Kinect V2 のデータと座標系

座標系	Kinect V2 のデータ
ColorSpace	カラー画像
DepthSpace	深度画像，BOdyIndex 画像，赤外線画像
CameraSpace	骨格情報

大きいけれども必ずしも平易とはいいがたい．

3. NtKinect クラス・ライブラリ

Kinect V2 の機能を利用したプログラミング開発をできるだけ見通しよく行うために，C++ のクラス・ライブラリ NtKinect [4] を新しく開発した．

開発の基本方針は次の通りであり，NtKinect = Kinect SDK + オブジェクト + STL(C++) + OpenCV + mutex である．

- デバイス全体をひとつのオブジェクトで管理する
- Collection データの管理に STL を用いる
- OpenCV のデータ構造を利用する
- マルチ・スレッドに対応する
- DLL 化を容易にする

3.1 オブジェクトによるデバイスの管理

Kinect デバイスが持つセンサーを全てひとつのオブジェクトで管理する．一度のメソッド呼び出しで指定したセンサーからデータを取得しオブジェクト内に保存する．各センサーの初期化は，データを取得する最初のメソッド呼び出しで行う．

3.2 C++と STL の活用

Kinect V2 によって認識される人間のデータ数は刻々と変化する．そのようなデータは Collection として，STL を用いて管理する．

取得されたデータはヒープ上に割り当てるので，作成したプログラムの DLL 化が容易になる．

3.3 OpenCV のデータ構造の利用

画像処理や画像認識のプログラミングおいて OpenCV は有用であり，特に version 2 以降からは C++ で利用しやすくなった．Kinect V2 から取得されたデータを OpenCV のデータ構造を用いてデータを管理すれば，そのまま OpenCV の関数に渡せるのでその利点は大きい．

カラー画像，深度 (距離) 画像，BodyIndex 画像，赤外線画像を OpenCV の cv::Mat データ構造で表現する．各センサーごとに座標系が異なるので，必要に応じて座標変換を行う手段を用意する．(図 1)

3.4 マルチスレッド対応

センサーデバイスの制御と，結果データの処理は，平行

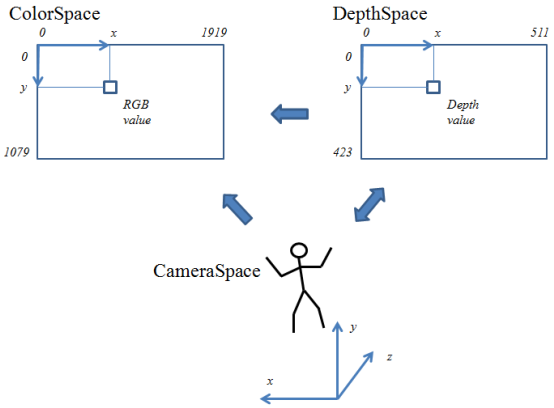

図 1　cv::Mat によるデータ表現と座標変換

動作をしないとプログラム全体の実行効率が悪くなる．そこで，mutex を用いた排他制御を行うメソッドと，スレッドセーフなメソッドを用意する．

3.5 DLL 化

上記の方針を採用したことで，NtKinect オブジェクト自体や Kinect センサーで取得したデータをヒープ上で管理できるので，開発したプログラムを DLL 化することが可能になる．DLL 化すれば他の言語や開発環境に組み込んで利用できる．

4. NtKinect の API

第 3 節で述べた設計方針に基いて C++ を用いて実装した．

メソッドの API を表 3 に，影響を受けるメンバ変数を表 4 に示す．表中ではメソッドの仮引数および一部のメンバ変数の型については省略している．

基本的に，一度のメソッド呼し出しでオブジェクト内の変数にデータが格納される，という状況を繰り返すことを想定している．データはオブジェクト外から変数にアクセスすることで参照する．

現在のバージョンでは，Kinect V2 の骨格認識，顔認識，詳細な顔認識 (HDFace)，ジェスチャ認識，音声の取得，音声方向の取得，音声認識に対応している．

5. NtKinect を用いたプログラムの開発

骨格認識と顔認識 を行う C++のコードを図 2 に，実行結果を図 3 に示す [5]．

プログラムの動作は以下の通りである．

- kinect.setSkeleton() メソッドを呼び出して骨格認識を行う．

表 3　NtKinect のメソッド
Table 3　Methods of NtKinect.

機能	返り値型	メソッド名	説明 (影響を受けるメンバ変数)
RGB カメラ	void	setRGB	カラー画像を取得する (rgbImage)
深度	void	setDepth	深度画像を取得する (depthImage)
BodyIndex	void	setBodyIndex	骨格インデックス画像を取得する (bodyIndexImage)
赤外線	void	setInfrared	赤外線画像を取得する (infraredImage)
骨格認識	void	setSkeleton	骨格を認識する (skeleton, skeletonId, skeletonTrackingId)
手の平認識	pair<int,int>	handState	手の平の状態を認識する ()
顔認識	void	setFace	顔を認識する (facePoint, faceRect, faceDirection, faceProperty)
顔認識	void	setHDFace	顔を詳細認識する (hdfaceVertices, hdfaceTrackingId, hdfaceStatus)
音声	void	setAudio	音声を取得する (beamAngle, beamAngleConfidence, audioTrackingId)
音声	void	drawAudioDirection	音声ビームの方向を描画 ()
音声	bool	isOpendAudio	録音中かどうかを返す ()
音声	void	opendAudio	録音を開始する ()
音声	void	closeAudio	録音を終了する ()
音声認識	void	setSpeechLang	音声認識の言語と単語ファイルを設定する ()
音声認識	void	startSpeech	音声認識を開始する ()
音声認識	void	stopSpeech	音声認識を終了する ()
音声認識	bool	setSpeech	音声を認識する (recognizedSpeech, speechTag, speechItem, speechConfidence)
ジェスチャ認識	void	setGestureFile	ジェスチャ定義ファイルを設定する
ジェスチャ認識	void	setGesture	ジェスチャを認識する (discreteGesture, discreteGestureTrackingId, continuousGesture, continuousGestureTrackingId)

表 4　NtKinect のメンバ変数
Table 4　Member variables of NtKinect.

機能	型	変数名	説明
RGB カメラ	cv::Mat	rgbImage	カラー画像
深度カメラ	cv::Mat	depthImage	深度画像
BodyIndex	cv::Mat	bodyIndexImage	BodyIndex 画像
赤外線	cv::Mat	infraredImage	赤外線画像
骨格認識	vector<vector<Joint>>	skeleton	骨格情報
骨格認識	vector<int>	skeletonId	骨格の BodyIndex
骨格認識	vector<INT64>	skeletonTrackingId	骨格の TrackingID
顔認識	vector<vector<PointF>>	facePoint	顔の部品の位置
顔認識	vector<cv::Rect>	faceRect	顔の矩形領域
顔認識	vector<cv::Vec3f>	faceDirection	顔の向き
顔認識	vector<vector<DetectionResult>>	faceProperty	顔の状態
顔認識	vector<UINT64>	faceTrackingID	顔の skeletonTrackingID
顔詳細認識	vector<vector<CameraSpacePoitn>>	hdfaceVertices	顔の部品の座標
顔詳細認識	vector<UINT64>	hdfaceTrackingId	顔の skeletonTrackingId
顔詳細認識	vector<pair<int,int>>	hdfaceStatus	顔の詳細認識の状態
音声	float	beamAngle	音声の方向
音声	float	beamAngleConfidence	beamangle の信頼度
音声	UINT64	audioTrackingId	発話者の skeletonTrackingId
音声認識	bool	recognizedSpeech	音声認識できたかのフラグ
音声認識	wstring	speechTag	音声認識できた単語のタグ
音声認識	wstring	speechItem	音声認識できた単語の項目
音声認識	float	speechConfidence	音声認識の精度
音声認識	float	confidenceThreshold	音声認識の閾値
ジェスチャ認識	vector<pair<...,float>>	discreteGesture	認識したジェスチャと信頼度
ジェスチャ認識	vector<UINT64>	discreteGestureTrackingId	ジェスチャの skeletonTrackingId
ジェスチャ認識	vector<pair<...,float>>	continuousGesture	認識したジェスチャと進捗度
ジェスチャ認識	vector<UINT64>	continuousGestureTrackingId	ジェスチャの skeletonTrackingId

```
#define USE_FACE
#include "NtKinect.h"
using namespace std;
void doJob() {
  NtKinect kinect;
  while (1) {
    kinect.setRGB();
    kinect.setSkeleton();
    kinect.setFace();
    for (auto& r : kinect.faceRect)
      cv::rectangle(kinect.rgbImage, r, cv::Scalar(0, 0, 0), -1);
    for (auto& person : kinect.skeleton)
      for (auto& joint : person) {
        if (joint.TrackingState == TrackingState_NotTracked) continue;
        ColorSpacePoint cp;
        kinect.coordinateMapper->MapCameraPointToColorSpace(joint.Position,&cp);
        cv::rectangle(kinect.rgbImage,cv::Rect((int)cp.X-5,(int)cp.Y-5,10,10),cv::Scalar(0,0,255)
    ,2);
      }
    cv::imshow("rgb", kinect.rgbImage);
    auto key = cv::waitKey(1);
    if (key == 'q') break;
  }
  cv::destroyAllWindows();
}
int main(int argc, char** argv) {
  try {
    doJob();
  } catch (exception &ex) {
    cout << ex.what() << endl;
  }
  return 0;
}
```

図 2 骨格認識と顔認識の C++コード

図 3 骨格認識と顔認識の例

- kinect.setFace() メソッドを呼び出して顔認識する．
- 顔の矩形領域を黒で塗り潰す．
- 関節の位置は CameraSpace 座標系で表現されているので，ColorSpace 座標系の座標に変換してから RGB 画像上に四角形を描画する．
- Joint 型のデータは，その TrackingState メンバ変数の値が TrackingState_NotTracked である場合は意味を持たないのでその場合は関節の表示処理を省略する．

6. NtKinect を用いた DLL ファイルの開発

顔認識を行い，最大 6 人までの顔の方向 (Euler 角, すなわち pitch, yaw, roll の角度) を返す DLL のヘッダファイルを図 4 に，C++のコードを図 5 に示す．

DLL プログラムを開発する場合は，NtKinect 型のデータをヒープ上に確保し，void *型のデータとして返す．また，センサーから取得した Collections データは，DLL 関数を呼び出す側が確保したメモリ領域にコピーすることで値を返す [6]．

```
#ifdef NTKINECTDLL_EXPORTS
#define NTKINECTDLL_API __declspec(dllexport)
#else
#define NTKINECTDLL_API __declspec(dllimport)
#endif
extern "C" {
  NTKINECTDLL_API void* getKinect(void);
  NTKINECTDLL_API int faceDirection(void* ptr, float* dir);
}
```

図 4　顔認識 DLL の C++ヘッダ

```
#include "stdafx.h"
#include "NtKinectDll.h"
#define USE_FACE
#include "NtKinect.h"
using namespace std;
NTKINECTDLL_API void* getKinect(void) {
  NtKinect* kinect = new NtKinect;
  return static_cast<void*>(kinect);
}
NTKINECTDLL_API int faceDirection(void* ptr,float *dir) {
  NtKinect *kinect = static_cast<NtKinect*>(ptr);
  (*kinect).setSkeleton();
  (*kinect).setFace();
  int idx=0;
  for (auto d: (*kinect).faceDirection) {
    dir[idx++] = d[0];
    dir[idx++] = d[1];
    dir[idx++] = d[2];
  }
  return (*kinect).faceDirection.size();
}
```

図 5　顔認識 DLL の C++コード

7. DLL ライブラリの他のプログラミング言語や開発環境からの利用

第 6 節で示した DLL プログラムを，Unity で利用するコードを図 6 に示す．Unity のプロジェクトの Assets/x86_64/ フォルダの下に DLL ファイルなどの必要なファイルを置くと利用できる [6] .

実行すると，認識した顔の個数だけ立方体が赤くなって顔の向きに合わせて回転する (図 7).

8. 考察

Kinect V2 を C++ からプログラミングするためのクラス・ライブラリである NtKinect について，その設計方針を論じた．Kinect V2 の機能を活用するプログラムが簡潔に記述でき，またその DLL 化および他言語や開発環境からの利用も容易なことからその有用性は示されたと考える．

また Unity からの Kinect V2 の利用について，Microsoft は公式に対応したアセットを配布しているが，骨格認識・顔認識・ジェスチャ認識に対応するのみで，音声や音声方向の取得および音声認識には対応していない．NtKinect を用いると，音声や音声方向の取得および音声認識 [7] を含む Kinect V2 の全機能を DLL 化して Unity から利用可能となることから，NtKinect の方に優位性があるといえる．

NtKinect を 2016 年 7 月にオープンソースとして公開して以来，[8], [9], [10] をはじめとして使用例は増えている．また，検索エンジンにおける NtKinect のページランクも上昇しており，2017 年 2 月 6 日現在 Google における NtKinect のページランクは表 5 に示すように上位である．

今後は他の言語や開発環境への対応を進めていく予定である．Unity への対応方法は既に Web で公開しているが，今後，特に Unreal Engine への対応を進めたい．

```
using UnityEngine;
using System.Collections;
using System.Runtime.InteropServices;
public class NtKinectBehaviour : MonoBehaviour {
  [DllImport ("NtKinectDll")] private static extern System.IntPtr getKinect();
  [DllImport ("NtKinectDll")] private static extern int faceDirection(System.IntPtr kinect,
    System.IntPtr data);
  private System.IntPtr kinect;
  void Start () {
    kinect = getKinect();
    ...
  }
  void Update () {
    float[] data = new float[bodyCount * 3];
    GCHandle gch = GCHandle.Alloc(data,GCHandleType.Pinned);
    int n = faceDirection(kinect,gch.AddrOfPinnedObject());
    gch.Free();
    for (int i=0; i<n/3; i++) {
      // data[i*3], data[i*3+1], data[i*3+2] = pitch, yaw, roll
    }
}
```

図 6 顔認識 DLL を呼び出す Unity C#コード (抜粋)

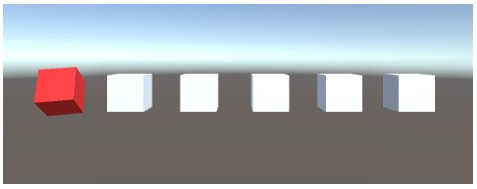

図 7 Unity からの DLL の利用例

表 5 Google における NtKinect のページランク

検索語	ページランク
骨格認識	1 位
骨格認識 kinect	1 位
顔認識 kinect	4 位
ジェスチャ認識 kinect	3 位

参考文献

[1] Microsoft: *Kinect for Windows* の紹介, https://developer.microsoft.com/ja-jp/windows/kinect/, (2017/02/03 access).
[2] Microsoft: *Kinect for Windows SDK C++ Reference*, https://msdn.microsoft.com/ja-jp/library/dn791993.aspx , (2017/02/03).
[3] 中村, 杉浦, 高田, 上田：KINECT for Windows SDK プログラミング v2 センサー対応版，秀和システム (2015).
[4] Yoshihisa Nitta: *NtKinect - Kinect V2 C++ Programming with OpenCV on Windows10*, http://nw.tsuda.ac.jp/lec/kinect2/ (2017/02/03 access).
[5] Yoshihisa Nitta: NtKinect- Kinect V2 を顔を認識する (ColorSpace 座標系), http://nw.tsuda.ac.jp/lec/kinect2/KinectV2_face/ (2017/02/03 access).
[6] Yoshihisa Nitta: NtKinect- Kinect V2 を用いた顔認識を DLL 化して Unity から利用する, http://nw.tsuda.ac.jp/lec/kinect2/KinectV2_dll2/ (2017/02/03 access).
[7] Yoshihisa Nitta: NtKinect- Kinect V2 を用いた音声認識を DLL 化して Unity から利用する, http://nw.tsuda.ac.jp/lec/kinect2/KinectV2_dll3/ (2017/02/08 access).
[8] 土屋, 新田: 人物認識によるインタラクティブ・ライトアートシステム, 映像表現・芸術科学フォーラム 2016, P-131 (2016).
[9] 土屋, 伊藤, 新田: 人物認識によるインタラクティブ・ペンライトアートシステム, NICOGRAPH 2016, P-4 (2016).
[10] 土屋, 伊藤, 新田: 人物シルエットをペンライトアート風に表現するインタラクティブシステム, WISS 2016, P-213 (2016).