Exemple de code pour la caméra de détection de profondeur Intel® RealSense™ R200

Suivi de visage

Par Bryan Brown (Intel)

Date de publication : 31 octobre 2015

Vous pouvez commenter cet exemple de code sur le forum : **Commentez**

I - Introduction....3
II - Environnement de développement logiciel....3
II-A - Description du matériel....3
III - Notes de version....4
IV - À propos du code....4
V - Ressources....4
VI - À propos de l'auteur....5

Les sources présentées sur cette page sont libres de droits et vous pouvez les utiliser à votre convenance. Par contre, la page de présentation constitue une œuvre intellectuelle protégée par les droits d'auteur. Copyright ® 2015 Bryan Brown (Intel). Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.
https://intel.developpez.com/tutoriels/realsense/r200/exemple-code-camera-profondeur-suivi-visage/

I - Introduction

Cet exemple de code C#/XAML donne les bases d'utilisation de l'algorithme de suivi de visage dans le **SDK Intel® RealSense™ pour Windows*** pour détecter et suivre le visage d'une personne en temps réel en utilisant la caméra R200. L'exemple de code réalise les fonctions suivantes :

- afficher le flux de couleurs en temps réel de la caméra RGB R200 dans un champ image ;
- superposer un rectangle de contrôle qui suit le visage de l'utilisateur (selon son apparence dans la scène) ;
- afficher le nombre de visages détectés par la caméra R200 ;
- afficher la hauteur et la largeur du visage suivi ;
- afficher les coordonnées en deux dimensions (X et Y) du visage suivi ;
- indiquer la profondeur du visage, soit la distance entre le visage et la caméra R200 ;
- activer et afficher une surveillance des alertes et utiliser un gestionnaire d'événements d'alerte.

Illustration 1. Exemple de code de suivi de visage

II - Environnement de développement logiciel

L'exemple de code a été créé sous Windows® 10 RTM avec Microsoft Visual Studio* Community 2015. Le modèle de projet utilisé pour cet exemple est **Visual C#** - **Windows** - **Classic Desktop**.

Les versions SDK et DCM utilisées dans ce projet sont :

- Intel® RealSense™ SDK v6.0.21.6598
- Intel® RealSense™ Depth Camera Manager R200 v2.0.3.39488

II-A - Description du matériel

Pour ce développement, nous avons utilisé le **Kit Développeur Intel® RealSense™ (R200)** avec la caméra installée sur un trépied standard utilisant un support magnétique en option ** (Figure 2).

Illustration 2. Caméra installée sur un support magnétique en option **

(** Peut ne pas être disponible dans tous les kits.)

Les prérequis matériels pour faire tourner l'exemple de code R200 sont les suivants :

- processeur Intel® Core™ de 4e génération (ou plus récent) ;
- 150 Mo d'espace disque libre ;
- 4 Go de RAM ;
- caméra Intel® RealSense™ (R200) ;
- port USB 3 disponible pour la caméra R200 (ou connexion dédiée pour les caméras intégrées).

Note importante : *une interface USB 3 est requise pour prendre en charge la bande passante requise par la caméra. Cette interface doit être connectée à un port USB 3 dédié sur le système du client (ne pas utiliser de hub).*

Les sources présentées sur cette page sont libres de droits et vous pouvez les utiliser à votre convenance. Par contre, la page de présentation constitue une œuvre intellectuelle protégée par les droits d'auteur. Copyright ® 2015 Bryan Brown (Intel). Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.
https://intel.developpez.com/tutoriels/realsense/r200/exemple-code-camera-profondeur-suivi-visage/

III - Notes de version

- Le projet utilise l'espace de nom System.Drawing.Imaging, qui est référencé manuellement dans un nouveau projet en faisant un clic droit sur **Références** dans l'explorateur de solutions puis en sélectionnant **Ajouter une référence…** pour ouvrir la fenêtre du gestionnaire de références. Ensuite, selectionnez **Assemblys, Framework** puis cherchez dans la liste l'entrée **System.Drawing**. Cochez la case, puis cliquez sur le bouton **OK**.
- Le projet utilise un chemin explicite vers libpxcclr.cs.dll (la DLL managée) situé ici : C:\Program Files (x86)\Intel\RSSDK\bin\x64. Vous devrez changer cette référence si le chemin d'installation de votre SDK est différent.
- Puisque ce projet utilise une version 64 bits de la DLL, vous devez vous assurer que « x64 » est bien spécifié dans le paramètre **Projet → Propriétés → Plateforme**.
- Le projet inclut une commande d'événement après la génération pour garantir que la DLL non managée (libpxccpp2c.dll) soit copiée dans le répertoire de sortie cible :

```
if "$(Platform)" == "x86" (copy /y "$(RSSDK_DIR)\bin\win32\libpxccpp2c.dll" "$(TargetDir)"
 ) else ( copy /y "$(RSSDK_DIR)\bin\x64\libpxccpp2c.dll" "$(TargetDir)" )
```

IV - À propos du code

L'exemple de code est construit selon la structure suivante :

- configurer la session, l'interface SenseManager, et le module de détection de visage ;
- lancer une tâche appelée Update, dans laquelle la boucle AcquireFrame-ReleaseFrame est traitée ;
- les actions suivantes ont lieu dans la boucle AcquireFrame-ReleaseFrame :
    - obtenir les données de l'image,
    - obtenir les données du module de détection de visage,
    - appeler la méthode Render pour mettre à jour l'interface ;
- ressources de rendu
    - produire l'image ;
- les actions suivantes ont lieu dans la méthode Render:
    - appeler la méthode ConvertBitmap pour convertir chaque image bitmap vers un type BitmapImage, qui est nécessaire pour afficher chaque image dans le contrôle d'image WPF,
    - mettre à jour l'interface en déléguant le travail au dispatcher associé avec la tâche de l'UI ;
- une méthode ShutDown est appelée dans le cas où les événements Window_Closing ou btnExit_Click surviennent. Les actions suivantes se produisent dans la méthode ShutDown:
    - arrêter la tâche Update,
    - supprimer les objets.

Essayez-le : suivez le **lien de téléchargement** pour obtenir le code et essayer cet exemple.

Pour en savoir plus sur **Intel RealSense**, rendez-vous sur la **Zone des Développeurs Intel** ou sur la **communauté Intel RealSense**.

Découvrez également le **SDK Intel RealSense** et le **forum dédié**.

V - Ressources

- **Intel RealSense Coding**

Les sources présentées sur cette page sont libres de droits et vous pouvez les utiliser à votre convenance. Par contre, la page de présentation constitue une œuvre intellectuelle protégée par les droits d'auteur. Copyright ® 2015 Bryan Brown (Intel). Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.
https://intel.developpez.com/tutoriels/realsense/r200/exemple-code-camera-profondeur-suivi-visage/

VI - À propos de l'auteur

Bryan Brown est un ingénieur d'applications logicielles dans la Division des Relations Développeurs d'Intel.

Les sources présentées sur cette page sont libres de droits et vous pouvez les utiliser à votre convenance. Par contre, la page de présentation constitue une œuvre intellectuelle protégée par les droits d'auteur. Copyright ® 2015 Bryan Brown (Intel). Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.
https://intel.developpez.com/tutoriels/realsense/r200/exemple-code-camera-profondeur-suivi-visage/