ENSIETA

ÉCOLE NATIONALE SUPÉRIEURE D'INGÉNIEURS

Robot suiveur d'objet coloré par webcam montée sur servomoteur

> Sommaire

1. **Principe**
2. **Constitution**
   a) Plateforme mécanique + moteurs
   b) Carte de puissance
   c) Carte d'interface
   d) PC embarqué
   e) Communication
   f) Webcam
   g) Servomoteur
3. **Programmation**
   a) Récupération et traitement des images de la webcam
   b) Contrôle des moteurs et servomoteurs via Labjack
4. **TP**

Principe

- Meute de robots qui se suivent en chaîne de manière autonome
- Seul le 1er robot est téléguidé
- Chaque robot a un objet rouge à l'arrière, visible par le robot suivant qui cherche à le suivre

Constitution

Plateforme mécanique + moteurs

- Exemple : plateforme 4WD1

Carte de puissance

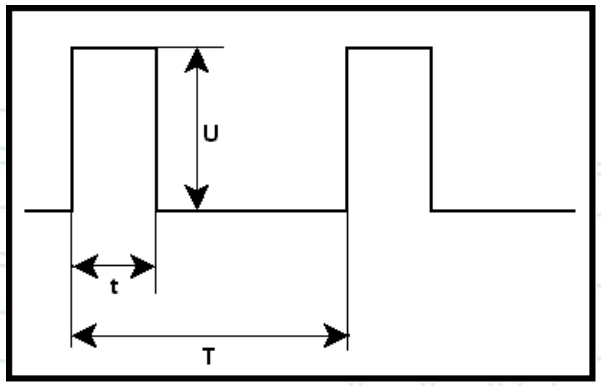
- Permet de contrôler les moteurs par des signaux de commande
  - Moteurs : tensions et courants élevés provenant des batteries
  - Signaux de commande : tensions et courants faibles venant directement ou indirectement du PC
    - Exemples : signaux PWM, I2C

Carte de puissance

- Exemple : Robbe Rokraft

Carte de puissance

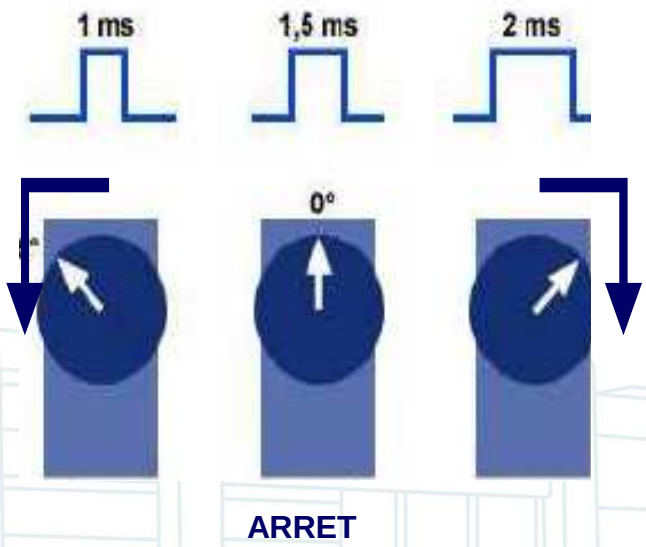
- Exemple : Robbe Rokraft
  - Fonctionnement
    - La puissance envoyée aux moteurs (et donc leur vitesse) dépend du signal de commande PWM
    - PWM = Pulse Width Modulation : modulation en largeur d'impulsion



U : tension du PWM (5 V)
t : largeur d'impulsion (entre 1 et 2 ms)
T : période (20 ms)

Carte de puissance

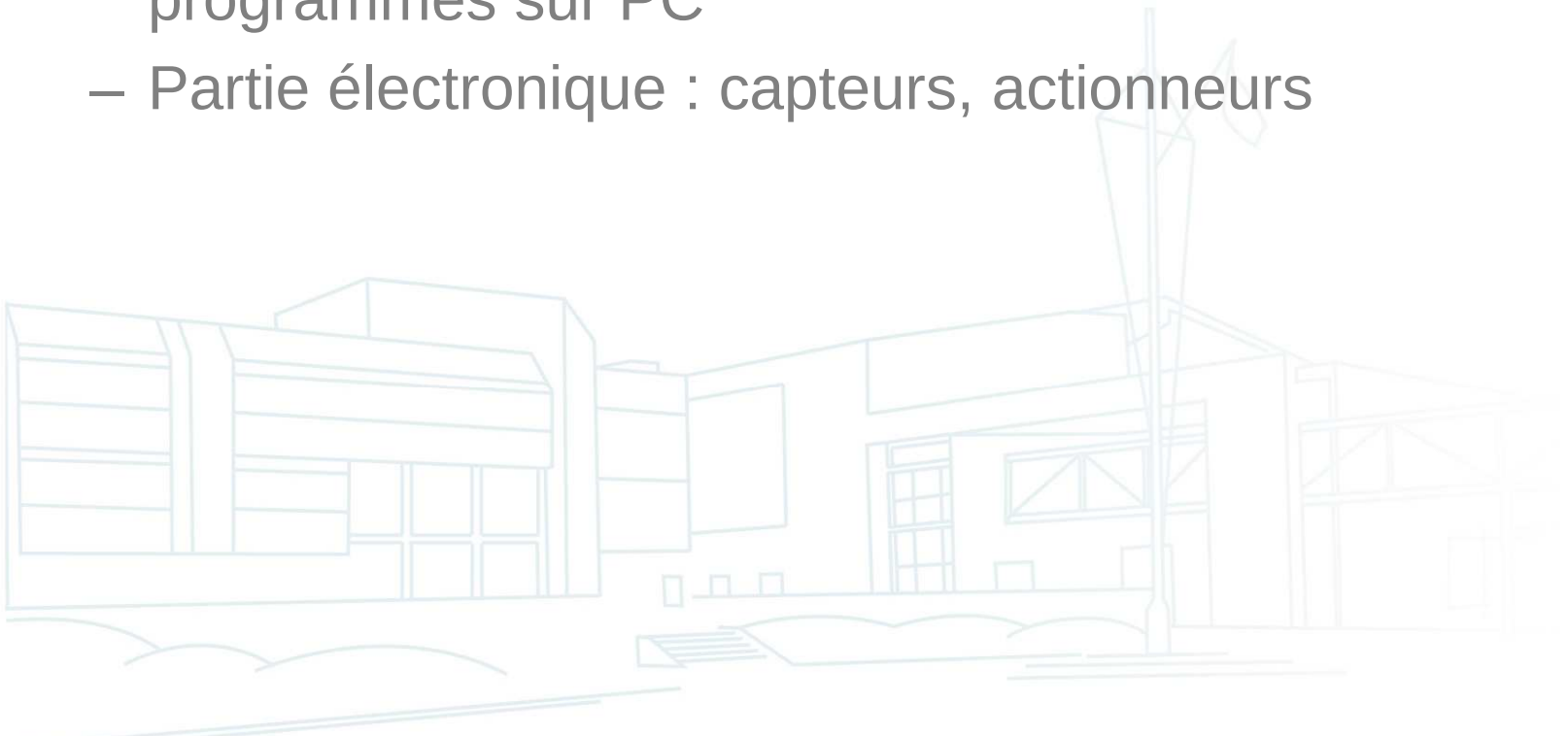
- Exemple : Robbe Rokraft
  - Fonctionnement
    - Correspondance largeur d'impulsion / vitesse de rotation



État du moteur	Largeur d'impulsion
Moteur à l'arrêt	**1.5 ms**
Rotation dans un sens, en accélérant	**1.5 à 2.0 ms**
Rotation dans le sens inverse, en décélérant	**1.0 à 1.5 ms**

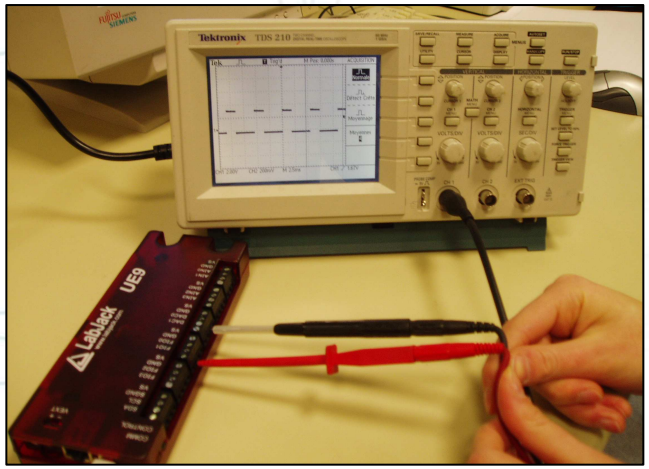
Carte d’interface

- Relie la partie informatique avec la partie électronique (capteurs, actionneurs)
  - Partie informatique : intelligence par le biais de programmes sur PC
  - Partie électronique : capteurs, actionneurs



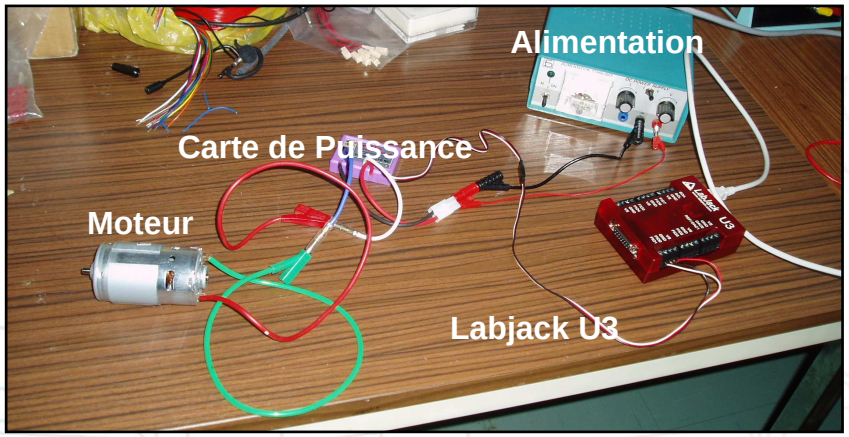
Carte d’interface

- Exemple : boitier Labjack
  - Se branche sur l'ordinateur en USB et est contrôlé par des programmes exécutés sur l'ordinateur
  - Peut générer des signaux PWM, I2C
  - Peut générer des petites tensions
  - Peut lire des petites tensions (venant de capteurs analogiques tels que des télémètres, odomètres, boussoles...)
  - ...



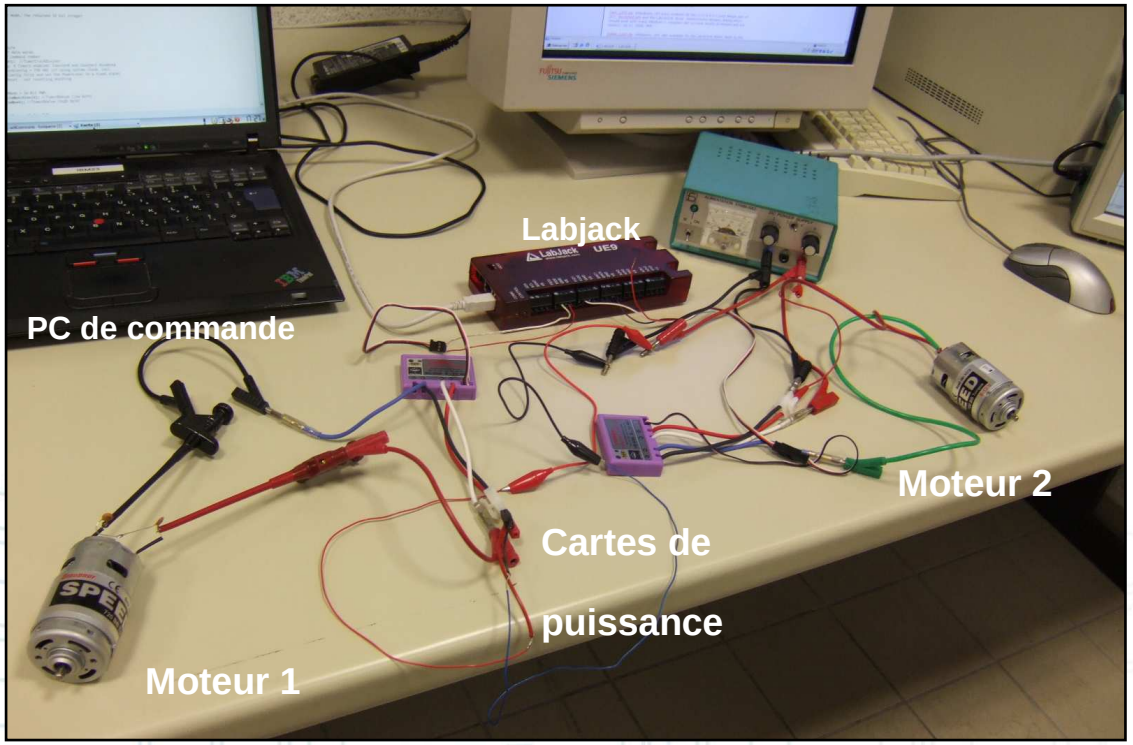
Carte d’interface

- Exemple : boitier Labjack



Carte d’interface

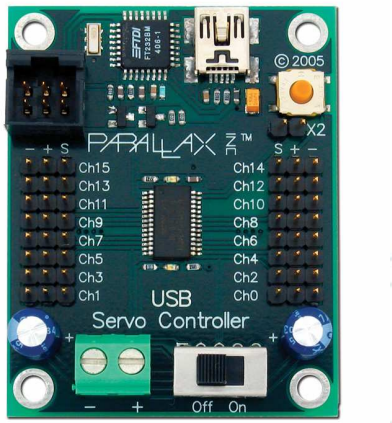
- Exemple : boitier Labjack



Dispositif de commande de moteurs avec la carte uE9

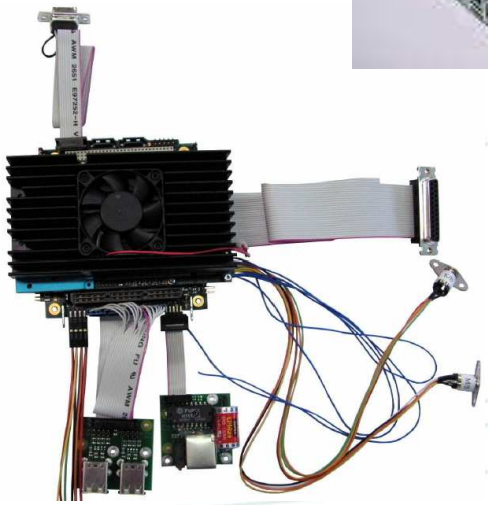
Carte d’interface

- Autres exemples : Cartes Parallax et Pololu
  - Se branchent en série (ou USB via un convertisseur USB-série) et génèrent jusqu’à 16 PWM



PC embarqué

- Intelligence du robot
  - Contient les programmes définissant le comportement du robot
- Exemples :
  - eeePC 901
  - Mini ITX
  - PC 104
  - …



Computer form factors	
Name	Size (mm)
eeePC 901	226 × 175.3 × 22.9
Mini-ITX	170×170
Nano-ITX	120×120
Pico-ITX	100×72
PC/104	96×90

Périphérique de communication

- Relie le robot au PC de commande
- Exemple : clé Wifi USB

Webcam

- Capteur que l'on peut simplement brancher sur un PC via USB
- Exemple : Logitech Quickcam Pro 9000
  - Images jusqu’à 1600 x 1200 pixels

Servomoteur

- Servomoteur = petit moteur + carte de puissance : pour orienter la webcam
- Commandé par PWM
- 2 types de servomoteurs :
  - Asservis en position : tournent de -40 à +40° par exemple
  - Asservis en vitesse

Programmation

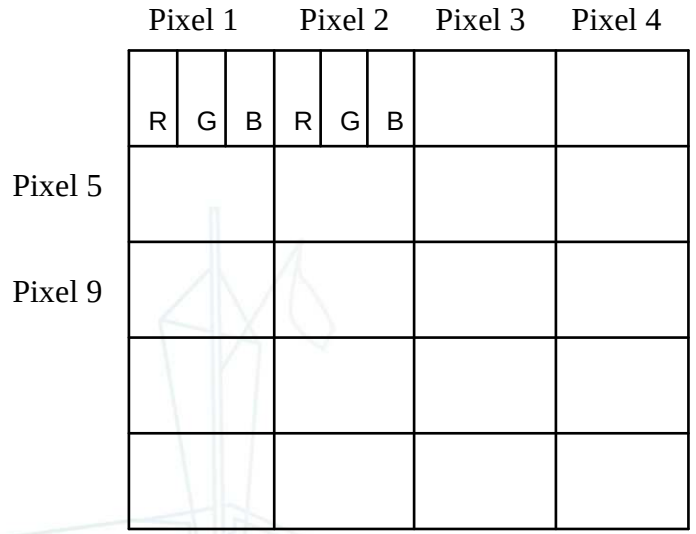
Récupération et traitement des images de la webcam

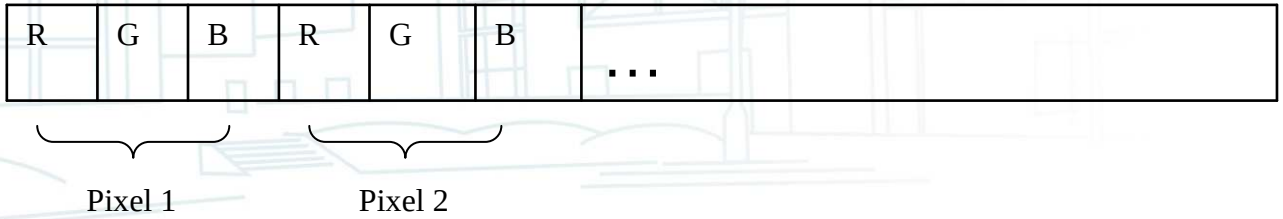
- Représentation courante d'une image en C/C++

```
unsigned char * img;
int width;
int height;

imgsize = width * height * 3;


// Pixel à ligne i, colonne j
img[0+3*j+3*width*i] = 0; // Rouge
img[1+3*j+3*width*i] = 0; // Vert
img[2+3*j+3*width*i] = 255; // Bleu
```





Récupération et traitement des images de la webcam

- Utilisation d'OpenCV
  - Présentation
    - http://opencv.willowgarage.com/wiki/
    - Bibliothèque open source
    - Portable
    - Fonctions en C/C++ ou Python
    - Codes optimisés développés à l'origine par Intel
    - Documentation et exemples de codes
    - …

Récupération et traitement des images de la webcam

- Utilisation d'OpenCV
  - Présentation
    - 4 parties principales :
      - CXCORE (types et fonctions de base : matrices, images, arbres, graphes, fonctions mathématiques, dessin de formes…)
      - CV (traitement d'image : détection d'objets, de mouvement, calibration…)
      - HIGHGUI (récupération et affichage d'images : lecture/enregistrement de fichiers images et videos, gestion des webcams, affichage dans des interfaces graphiques…)
      - MLL (arbres de décision, réseaux de neurones…)

Récupération et traitement des images de la webcam

- Utilisation d'OpenCV
  - Type IplImage

```
typedef struct _IplImage
{
    int  nChannels;     /* 3 composantes pour une image couleur (rouge, vert, bleu) */
    int  depth;         /* 8 bits (1 octet, taille d'un char) pour chaque composante */
    int  width;         /* Largeur de l'image en pixels */
    int  height;        /* Hauteur de l'image en pixels */
    char *imageData;  /* Pointeur vers les données de l'image */
    ...
}
IplImage;
```

  - Création d'une image couleur vierge

```
IplImage* im = cvCreateImage(
    cvSize(320, 240), // Taille de l'image
    8, // 8 bits (1 octet, taille d'un char) pour chaque composante
    3 // 3 composantes pour une image couleur (rouge, vert, bleu)
    );
```

Récupération et traitement des images de la webcam

- Utilisation d'OpenCV
  - Accès aux pixels d'une IplImage

```
IplImage* frame;

...

unsigned char* data = reinterpret_cast<unsigned char*>(frame->imageData);

for (int i = 0; i < frame->height; i++)
{
    for (int j = 0; j < frame->width; j++)
    {
        data[0+3*j+3*frame->width*i] = 0; // Bleu
        data[1+3*j+3*frame->width*i] = 0; // Vert
        data[2+3*j+3*frame->width*i] = 255; // Rouge
    }
}
```

Récupération et traitement des images de la webcam

- Utilisation d'OpenCV
  - Chargement et affichage d'un fichier image

```
// Chargement d'une image au format bitmap
IplImage* image = cvLoadImage("image.bmp");

// Création d'une fenêtre
cvNamedWindow("Fenêtre");

// Affichage de l'image
cvShowImage("Fenêtre", image);

// Attend que l'utilisateur appuye sur une touche
cvWaitKey(0);

// Libération de la mémoire utilisée pour l'image chargée
cvReleaseImage(&image);

// Destruction de la fenêtre
cvDestroyWindow("Fenêtre");
```

Récupération et traitement des images de la webcam

- Utilisation d'OpenCV
  - Récupération d'une image webcam

```
CvCapture* pCaptureFromCAM = cvCaptureFromCAM(0);

frame = cvQueryFrame(pCaptureFromCAM);

...

cvReleaseCapture(&pCaptureFromCAM);
```

Contrôle des moteurs et servomoteurs via Labjack

- http://www.labjack.com/
- Une bibliothèque de fonctions et drivers fournis pour différents langages, OS
- Programmes exemples disponibles
- Pour contrôler les moteurs, nous utiliserons les fonctions « timer » du Labjack

Contrôle des moteurs et servomoteurs via Labjack

- Le Labjack peut générer jusqu'à 6 timers/PWM dont la fréquence est définie par

Fréquence finale du PWM
Doit être proche de 1/20ms
pour pouvoir contrôler un servo

Fréquence interne du Labjack

$$f_{PWM} = \frac{f_{syst}}{timer\_clock\_divisor * 2^{16}}$$

Paramètre pouvant être propre à chaque PWM
pour faire varier sa fréquence indépendamment des autres

Mode du PWM
=> précision de la largeur d'impulsion

Contrôle des moteurs et servomoteurs via Labjack

- Fonctions utiles

```
LJ_HANDLE lngHandle;

// Ouvre le 1er Labjack U3. lngHandle contiendra son identifiant
OpenLabJack(LJ_dtU3, LJ_ctUSB, "1", 1, &lngHandle);

// Reset
ePut(lngHandle, LJ_ioPIN_CONFIGURATION_RESET, 0, 0, 0);
```

Contrôle des moteurs et servomoteurs via Labjack

- Fonctions utiles

```
// Activation des timers et configuration de leurs fréquences
alngEnableTimers[0] = 1; // Active Timer0 (il sera sur FIO4, voir lngTCPinOffset)
alngEnableTimers[1] = 1; // Active Timer1 (il sera sur FIO5, voir lngTCPinOffset)
alngEnableCounters[0] = 0; // Désactive le 1er compteur
alngEnableCounters[1] = 0; // Désactive le 2ème compteur
lngTCPinOffset = 4;  // Offset indiquant que le 1er timer sera sur FIO4.
lngTimerClockBaseIndex = LJ_tc48MHZ_DIV; // Fréquence interne de 48 MHz avec gestion du lngTimerClockDivisor
                                         // pour pouvoir obtenir la fréquence de PWM voulue
lngTimerClockDivisor = 16; // La fréquence du PWM sera 45.78 Hz (voir formule)
alngTimerModes[0] = LJ_tmPWM16; // La largeur d'impulsion sera codée sur 16 bits :
alngTimerModes[1] = LJ_tmPWM16; // 0 -> largeur min, 65535 : largeur max
adblTimerValues[0] = 65535; // Largeur d'impulsion de départ de Timer0
adblTimerValues[1] = 0; // Largeur d'impulsion de départ de Timer1

eTCConfig(lngHandle, alngEnableTimers, alngEnableCounters, lngTCPinOffset, lngTimerClockBaseIndex,
    lngTimerClockDivisor, alngTimerModes, adblTimerValues, 0, 0);
```

Contrôle des moteurs et servomoteurs via Labjack

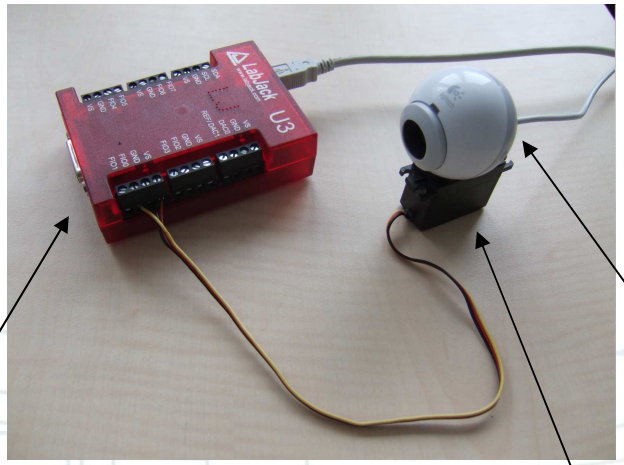
- Fonctions utiles

```
// Modification des largeurs d'impulsion
alngReadTimers[0] = 0;
alngReadTimers[1] = 0;
alngUpdateResetTimers[0] = 1; // Mettre à jour Timer0
alngUpdateResetTimers[1] = 0; // Ne pas modifier Timer1
alngReadCounters[0] = 0;
alngReadCounters[1] = 0;
alngResetCounters[0] = 0;
alngResetCounters[1] = 0;
adblTimerValues[0] = 32768; // Nouvelle largeur d'impulsion
adblTimerValues[1] = 0;

eTCValues(lngHandle, alngReadTimers, alngUpdateResetTimers, alngReadCounters,
    alngResetCounters, adblTimerValues, adblCounterValues, 0, 0);
```

TP

- Suivi d’un objet coloré à l’aide d’une webcam montée sur un servomoteur



Rappels C/C++

- Le C est inclus (à 99%) dans le C++ : quand on fait du C, on fait aussi du C++ mais l'inverse n'est pas forcément vrai
- Du code C ou C++ peut être écrit dans un fichier .cpp mais seul du code C peut être écrit dans un fichier .c
- Dans un .h, on peut écrire du C ou du C++, mais il faut que les .c n'incluent que des .h avec du C

Rappels C/C++

- Le C++ rajoute des notions de programmation orientée objet (classe, héritage, polymorphisme) ainsi que des facilités d’écriture

Rappels C/C++

- Différences entre les compilateurs Windows et Linux
  - Linux
    - Le compilateur C le plus utilisé est GCC
    - Son équivalent C++ est G++
  - Windows
    - GCC/G++ existent avec Cygwin et MinGW
    - Différents IDE existent et fournissent leurs propres compilateurs
      - Microsoft Visual Studio avec CL
      - Borland C++ Builder / Turbo C++ / Borland Developper Studio avec BCC32
      - Code Blocks / Dev-C++ avec MinGW

Rappels C/C++

- Différences entre les compilateurs Windows et Linux

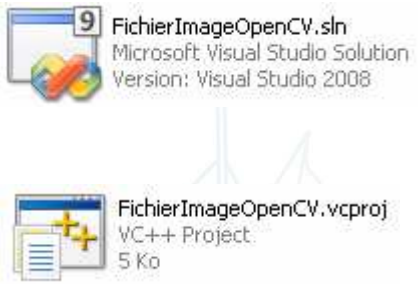
Equivalences Linux / Windows		
	Linux/GCC	Windows/Visual C++
Fichiers objets	.o	.obj
Bibliothèque statique	.a	.lib
Bibliothèque dynamique	.so	.dll
Exécutable	-	.exe

Rappels C/C++

- Visual Studio
  - Versions
    - Visual C++ 6 : date de 1998
    - Visual Studio 2002 / 2003 ou .Net : refonte de l'IDE et ajout des projets .Net
    - Visual Studio 2005 / 2008 : quelques mises à jour
  - Difficile à prendre en main au début : nombreux types de projets, nombreuses options incompréhensibles
  - Mais assez abouti, très utilisé, beaucoup d'aide possible sur Internet

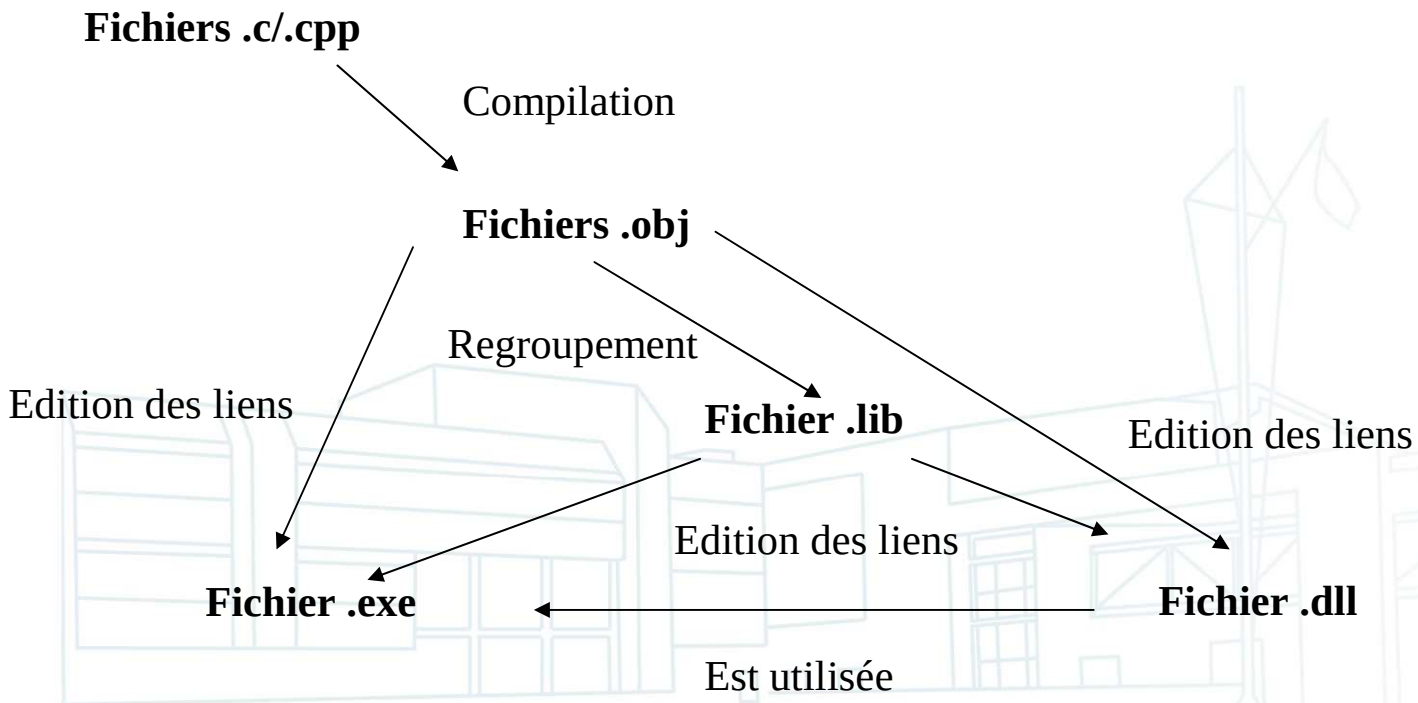
Rappels C/C++

- Visual Studio
  - Organisation
    - Workspace/Solution (fichier .dsw/.sln) : ensemble de projets
    - Projet (fichier .dsp/.vcproj) : ensemble de fichiers nécessaires à la génération d'un exécutable ou bibliothèque (.h, .c, .cpp…)



Rappels C/C++

- Utilisation de bibliothèques de fonctions externes



Utilisation d'OpenCV avec Visual C++ 6

- Les chemins suivants sont des chemins absolus considérant l'installation par défaut dans **C:\Program Files\OpenCV**

- Créer un projet **Win32ConsoleApplication**
- Dans le menu "**Project**", "**Settings**", "C/C++", categorie "**Preprocessor**".
  Ajouter les chemins suivants dans "**Additional include directories**":
  - **C:\Program Files\OpenCV\cv\include,**
  - **C:\Program Files\OpenCV\cvaux\include,**
  - **C:\Program Files\OpenCV\cxcore\include,**
  - **C:\Program Files\OpenCV\otherlibs\highgui**
- Dans le menu "**Project**", "**Settings**", "**Link**", catégorie "**Input**".
  Ajouter les librairies suivantes dans "**Object/library modules**":
  - **cv.lib cvaux.lib cxcore.lib highgui.lib**
- Ajouter le chemin suivant dans "**Additional library path**":
  - **C:\Program Files\OpenCV\lib**

- Modifier la variable d'environnement "PATH" de Windows en ajoutant:
  - **C:\Program Files\OpenCV\bin**

- Dans le code, ajouter
  - **#include "cvaux.h"**
  - **#include "highgui.h"**