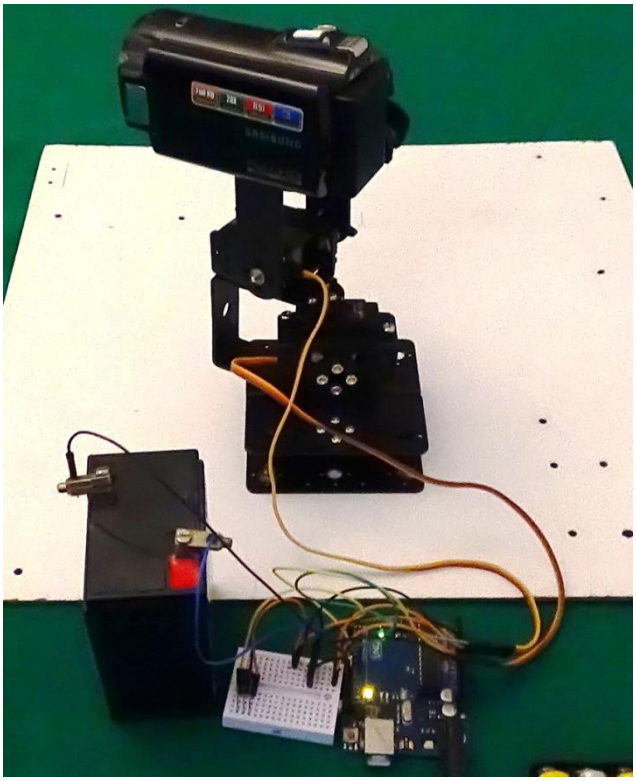
P – sistema di brandeggio telecomandato - pan&titl controlled by remote (some notes at end of this section)



Un sistema di brandeggio, pilotato tramite un telecomando, e' un comodo e sicuro sistema per orientare una telecamera, una macchina fotografica o un faretto collocati in un punto normalmente inaccessibile (come ad esempio una telecamera di sorveglianza, collocata sempre al di fuori della portata di coloro che vengono ripresi).

Il telecomando a infrarossi consente di controllare il sistema di brandeggio senza la necessita' di cavi, anche se in realta' e' piuttosto limitato poiche' la portata del telecomando non supera i 10 metri e comunque i due apparecchi (telecomando e cellula di rilevamento) devono essere in contatto "visivo".

Un muro o anche solo una tenda interrompe il contatto visivo e non ne consente il funzionamento.

Il problema puo' parzialmente essere superato utilizzando, al posto del sistema ad infrarossi, un sistema bluetooth o ancora meglio un sistema radiocomandato.

Prima di procedere all'eventuale costruzione del sistema e' necessario rilevare i codici del telecomando che si intende utilizzare (puo' essere utilizzato un qualunque telecomando di casa).

Tali codici devono poi essere inseriti nelle costanti "su", "giu", "dx" ed "sx" e "neutro" del programma.

Il "neutro" e' un valore (nel caso del telecomando usato in questo prototipo e' il valore "4294967295") che subentra al valore del tasto premuto nel caso in cui la pressione sia continua e prolungata (vedi anche i valori riportati nella prossima foto). Il programma, ovviamente, interpreta questo valore come una ripetizione del tasto premuto.

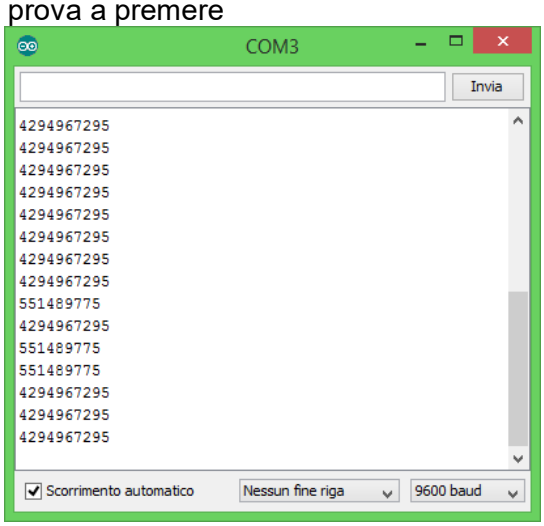
Per rilevare i codici dei tasti del telecomando si puo' utilizzare il seguente programma:

```
//----------------------programma per  rilevare i codici di un telecomando-------------------------
#include <IRremote.h>
int RECV_PIN = 5; // Pin di ricezione
IRrecv irrecv(RECV_PIN);

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Inizializzazione del ricevitore
}

void loop()
{
  decode_results results;
  if (irrecv.decode(&results))
  {
    // Tipi di codifica:
    //    results.decode_type == NEC
    //    results.decode_type == SONY
    //    results.decode_type == HEX
```

```
    //    results.decode_type == RC5
    //    results.decode_type == RC6
    if (results.decode_type == NEC)
       Serial.println(results.value);
    irrecv.resume(); // riceve il prossimo comando
  }
}
//-----------------------------------------------------------------------------------------------
```

I segnali emessi dai telecomandi appartengono a differenti famiglie di codifica (nec, sony, rc5, rc6) per cui, per prima cosa bisogna individuare la famiglia alla quale il telecomando appartiene.

Si compila quindi il programma utilizzando dapprima la famiglia SONY (sostituire SONY a NEC nella riga 19: "*if (result.decode_type==SONY)*") e poi si prova a premere qualche pulsante del telecomando. Se sul monitor seriale non compare nulla significa che non abbiamo indovinato la famiglia e quindi si ricompila e si riprova utilizzando un'altra famiglia di codifica. Se riprovando compare qualcosa significa che abbiamo azzeccato la famiglia e che ora possiamo individuare i codici che ci interessano.



Ogni volta che premiamo un pulsante sul monitor compaiono molti numeri, (vedi figura a lato). I numeri che ci interessano sono quelli che compaiono esattamente nel momento in cui premiamo il pulsante. Nel nostro esempio il codice che ci interessa e' il 551489775 mentre l'altro, che viene ripetuto in continuazione quando si tiene premuto un pulsante, e' quello da inserire nella costante "neutro" del programma. Una volta individuato il codice del tasto che ci interessa, lo trascriviamo per poterlo poi utilizzare nel programma di gestione del brandeggio.

Prima di procedere alla compilazione del programma deve essere installata, se non gia' presente, la seguente libreria:

- IRremote.zip, reperibile qui: https://www.pjrc.com/teensy/td_libs_IRremote.html

Per installare una libreria e' necessario seguire la procedura illustrata nei precedenti progetti, e sintetizzabile in:

- download della libreria in formato complesso
- installare la nuova libreria andando in IDE-> sketch-> includes Library-> add .zip library
- verificare l'avvenuta installazione (andando in IDE-> sketch-> includes Library-> Contributed library)

Nota: Questo esercizio e questa nota sono parte di una serie che vede protagonisti Arduino ed alcuni dei componenti ad esso collegabili. Per la maggior parte degli esercizi e' anche disponibile un filmato su youtube.

- Esercizi facenti parte della raccolta
- Filmati presenti su youtube
- Informazioni su arduino e sui componenti collegabili (PDF scaricato nell'area di download)
- Breve manuale di programmazione (PDF scaricato nell'area di download)

Per eventuali chiarimenti o suggerimenti sul contenuto di questa scheda scrivere a giocarduino@libero.it

Here some notes about this project, translated by google translator 

A pan & tilt system, controlled via a remote, is a safe and convenient system for orienting a camera placed in a normally inaccessible point (such as a surveillance camera, always placed outside the reach of those who are resumed).

An infrared remote control allows you to control the pan & tilt system without cables, although is rather limited because the scope does not exceed 10 meters and in any case the two devices (remote control and detection cell) must be in "visual" contact.
A wall or even just a curtain interrupts the visual contact and does not allow the operation. The problem can be partially overcome by using, in place of the infrared system, a Bluetooth system or even better a radio system.

If you want build the system you need to detect the remote control codes that you want to use (you can use any remote).

These codes must then be put in the program constants "su", "giu", "dx", "sx" and "neutro" (constants names means: up, down, right, left and neutral).

The “neutro” value (on my remote is "4294967295") is the value received when the pressure on a button is continuous and prolonged (see also the values shown in the second photo). The program, of course, interprets this value as a repetition of the key pressed.

To detect codes you can use the above program.

The remote controls belongs to different families (nec, sony, rc5, rc6) so, you need first locate the family to which belongs the remote control. Compiles the program using first the SONY family (insert SONY instead of NEC, in line 19: "if (result.decode_type == SONY)") and then press a button on remote.

If serial monitor is blank it means that you have not guessed the family. So you must recompile and retest using another coding family. If trying again appears something, means you guessed the family and you can now identify codes you are interested.

Each time you press a button, on monitor many numbers appear (see second figure). The code you are interested are those that appear exactly at the moment when you press the button. In our example, the code that we was looking for is 551489775 (the other codes are “neutral”- see above).

Before proceeding to program compilation must be installed, if not already done, the library:

- IRremote.zip, found here: https://www.pjrc.com/teensy/td_libs_IRremote.html

For library installation, see process shown in previous projects, and summarized in:

- library download in compressed form;
- Installation via IDE-> sketch-> includes Library-> add .zip library
- After installation please verify the library. It must be present in IDE-> sketch-> includes Library-> Contributed library

Note: This project and this note is part of a series that sees, as main characters, Arduino and some of connectable components. For most projects there is also a video on youtube.

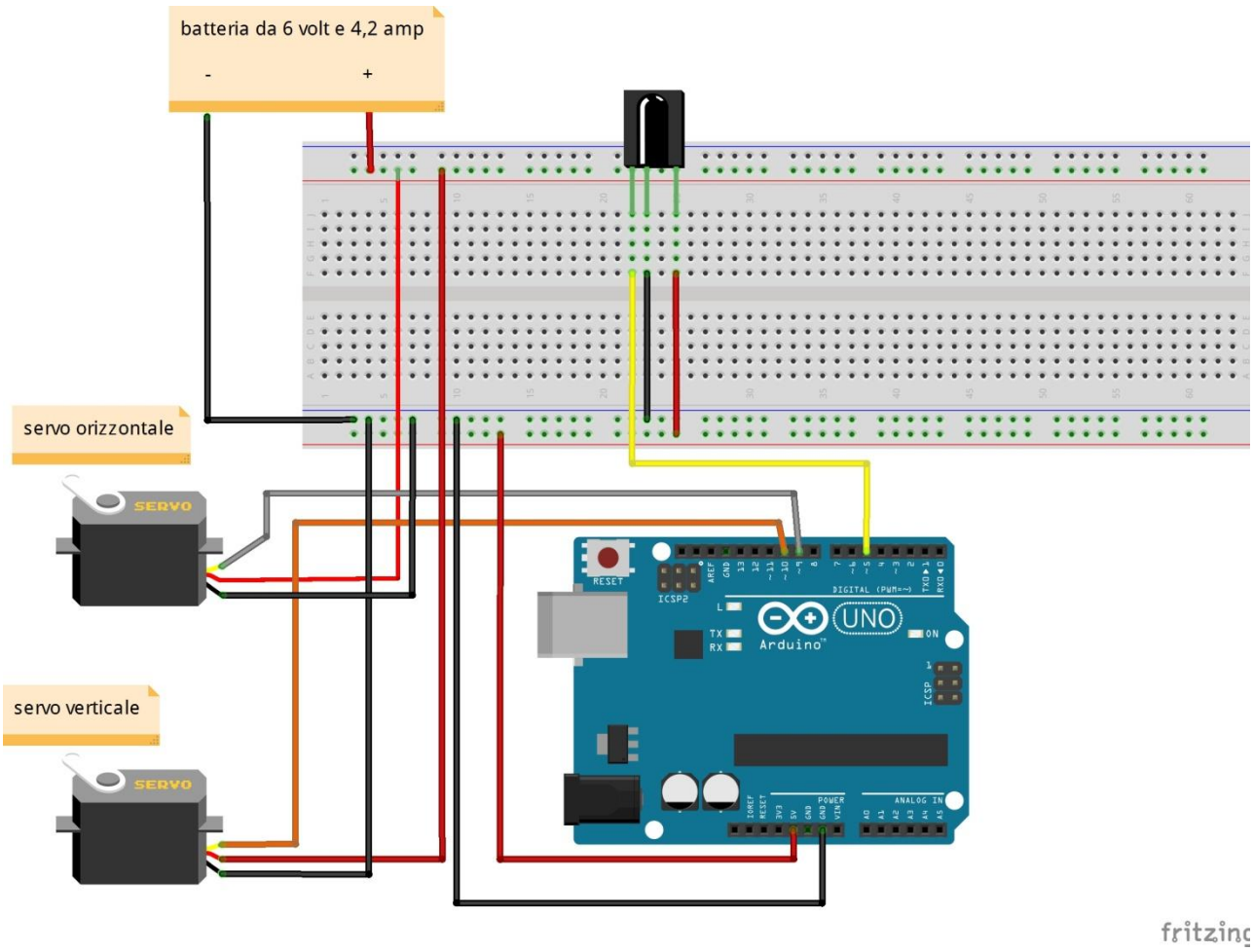
- Projects collection
- Movies on youtube
- About Arduino and components (italian; pdf will be downloaded in your download area
- Quick programming guide (almost english; pdf will be downloaded in your download area)

For any questions or suggestions about this note (and on its english translation), please write to giocarduino@libero.it (simple words and short sentences, please)

Materiali

- Due servo tipo TowerPro MG995
- Un ricevitore di segnali infrarossi
- Un qualunque telecomando di casa
- Due alloggi metallici per servomotori
- Due staffe ad “U” per servomotori
- Un po’ di cavetteria
- Una batteria da 6-7 volt e 4-6 ampere

Schema



Programma

```
/* Attenzione: facendo il copia/incolla dal PDF all'IDE si perde la formattazione del testo. Per
 * rendere piu' facilmente leggibile il programma e' opportuno formattarlo subito dopo il
 * trasferimento nell'IDE, premendo CTRL+T.
 *
 * Questo programma utilizza due servomotori, un ricevitore di segnali infrarossi ed un telecomando
 * per simulare un sistema di pan&tilt per il brandeggio di una telecamera.
 *
 *-------------------------------------------------------------------------------------------------
```

Arduino: brandeggio telecomandato - pan & tilt controlled by remote

```
 * Warning: cut&paste from PDF to IDE loses formatting. To restore it press CTRL + T.
 * This program uses two servo motors, an infrared receiver and a remote to control
 * a pan & tilt system for a camera.
 * ------------------------------------------------------------------------------------------
 */
#include <IRremote.h>  // richiama la libreria di gestione del telecomando
#include <Servo.h>  // richiama la libreria di gestione dei servomotori
Servo hor1;  // crea il servo oggetto "hor1" utilizzato per i movimenti orizzonatli.
Servo ver1;  // crea il servo oggetto "ver1" utilizzato per i movimenti verticali.
int RECV_PIN = 5; // Pin del ricevitore ad infrarossi
IRrecv irrecv(RECV_PIN); // definisce il ricevitore ad infrarossi
//
unsigned long su  = 16613503;  // codice del tasto su - remote code for up
unsigned long giu = 16617583;  // codice del tasto giu - remote code for down
unsigned long dx  = 16605343;  // codice del tasto dx - remote code for right move
unsigned long sx  = 16589023;  // codice del tasto sx -remote code for left move
unsigned long neutro = 4294967295; // ripete la precedente azione - remote code for action repeat
unsigned long lv  = 0;  // zona di lavoro - working area
unsigned long tempoattuale = 0;  // zona di memorizzazione del tempo corrente - current time
unsigned long tempoprec = 0; /* zona di memorizzazione del tempo trascorso dall'ultimo dei
servomotori   - time of last servo move*/
int hor  = 0; /* spostamento orizzontale; 1 grado per volta; positivo spostamento antiorario,
negativo spostamento orario */
int ver  = 0; /* spostamento verticale; 1 grado per volta; positivo spostamento verso il basso,
negativo spostamento verso l'alto */
int gradihor = 0;  // memorizzazione dei gradi (della posizione) orizzontale  - horizontal degrees
int gradiver = 0;  // memorizzazione dei gradi (della posizione) verticale   - vertical degrees
int limitesu = 50;  // max gradi raggiungibili nello spostamento verticale -  vertical limit
int limitesx = 170; // max gradi raggiungibili nello spostamento orizzontale  - horizontal limit
int ritardo = 10; /*valore di rallentamento del movimento dei servomotori  - Delay (10 microsecond)
before the next servo movement*/
//
// ********** muovi servomotori: routine di gestione dei servomotori
// ************** servo handling routine *************************
//
void muoviservomotori (void)
{
  gradiver = gradiver + ver;  /* aggiunge (o toglie, se ver e' negativo) un grado se e' stato
premuto un pulsante di movimento verticale */
  gradihor = gradihor + hor;  /* aggiunge (o toglie, se hor e' negativo) un grado se e' stato
premuto un pulsante di movimento orizzontale */
  if (gradiver  < 0)  // se tenta di andare oltre la posizione zero del servo verticale
    gradiver = 0;  // ripristina la poszione zero
  if (gradiver > limitesu) // se tenta di andare oltre il limite di spostamento verticale
    gradiver = limitesu;  // ripristina il limite verticale
  if (gradihor < 0) // se tenta di andare oltre il limite nello spostamento (orario) orizzontale
    gradihor = 0;  // ripristina la posizione zero
  if (gradihor > limitesx) // se tenta di andare oltre il limite (antiorario) orizzontale
    gradihor = limitesx;  // ripristina il limite di spostamento
  hor1.write(gradihor);  // indirizza il perno di hor1 alla posizione desiderata
  ver1.write(gradiver);  // indirizza il perno di ver1 alla posizione desiderata
}
//
//****  verifica telecomando: routine di ricezione e controllo codici da telecomando *******
// ************** check and receive signal from remote ********************************
//
void verifcatelecomando (void)
{
  decode_results results;
  if (irrecv.decode(&results))
  {
    if (results.decode_type == NEC)
    {
      lv = results.value;
      if (!(lv == neutro))
      {
        hor = 0;
        ver = 0;
        if (lv == su)
          ver = -1;
        if (lv == giu)
          ver = 1;
        if (lv == dx)
          hor = -1;
        if (lv == sx)
          hor = 1;
      }
    }
```

```
    irrecv.resume(); // resetta il ricevitore e lo prepara al prossimo comando
  }
}
//
//
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Inizializza il ricevitore
  hor1.attach(9);   // assegna il servo oggetto "hor1" (servo orizzontale) alla porta 9
  ver1.attach(10);   // assegna il servo oggetto "ver1" (servo verticale) alla porta 10
  hor1.write(gradihor);  // indirizza il perno di hor1 alla posizione iniziale (alla posizione zero)
  ver1.write(gradiver);  // indirizza il perno di ver1 alla posizione iniziale (alla posizione zero)
  tempoprec = millis ();  // memorizza il momento dell'ultimo movimento dei servo
}
//
//
void loop()
{
  verifcatelecomando ();
  tempoattuale = millis ();
  if (((tempoattuale - tempoprec) > ritardo) && (!(lv == 0))) /* se e' trascorso il tempo di
"ritardo" dall'ultimo movimento, ed e' pervenuto un segnale dal telecomando  -  if is elapsed
the "ritardo" time from last movement and received a command from remote */
  {
    tempoprec = tempoattuale;  // aggiorna tempoprec
    muoviservomotori ();
    lv = 0;   // azzera il segnale pervenuto dal telecomando
  }
}
```