Lire un capteur infrarouge avec Arduino

Yaug 27 juin 2014 Arduino, DIY, Domotique

Bonjour à tous,

J'ai reçu quelques capteurs intéressants, dont un capteur infrarouge (permettant par exemple de détecter une source de chaleur), on va donc reprendre un tuto avec ce capteur et un arduino (et un bonus en deuxième partie de tutoriel).

Le matériel

Pour ce tutoriel il nous faudra :

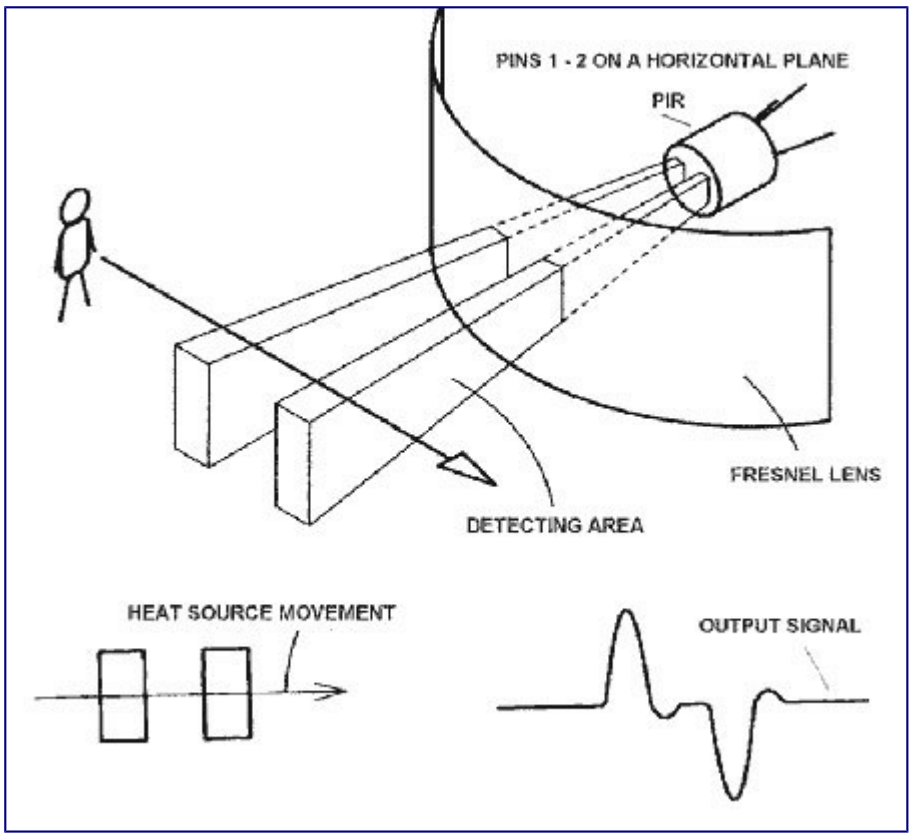
- Un arduino (ici un Uno)
- Un capteur HC SR-501
- 3 fils
- Une led 5mm

Et c'est tout. Minimaliste comme montage non ?

Le principe

Un capteur infrarouge permet de détecter un mouvement dans son champ de vision en se basant sur l'infrarouge. On parle aussi de capteur pyroélectrique ou PIR. Le PIR sont capable de détecter une variation des ondes infrarouges, ce qui génère un courant électrique. Dans le cas de notre capteur, il est en fait divisé en deux partie différente reliées ensemble afin de détecter une variation lors qu'une des moitiés capte plus qu'une autre. On a ainsi un relevé d'une différence, et non plus d'une valeur simple.

Lors d'un mouvement, la variation des deux moitiés vont varier, et on va donc capter cette variation positive.



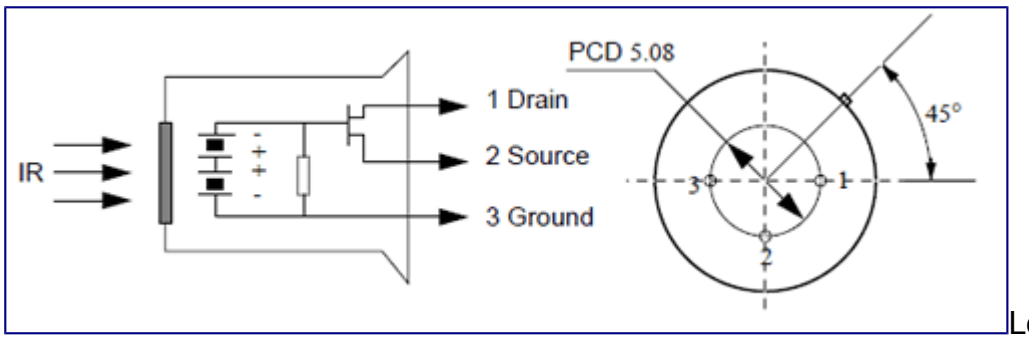
Principe de fonctionnement

Le capteur

Le capteur lui même ressemble à ça :

Le capteur

Et l’intérieur fonctionne de la manière suivante :

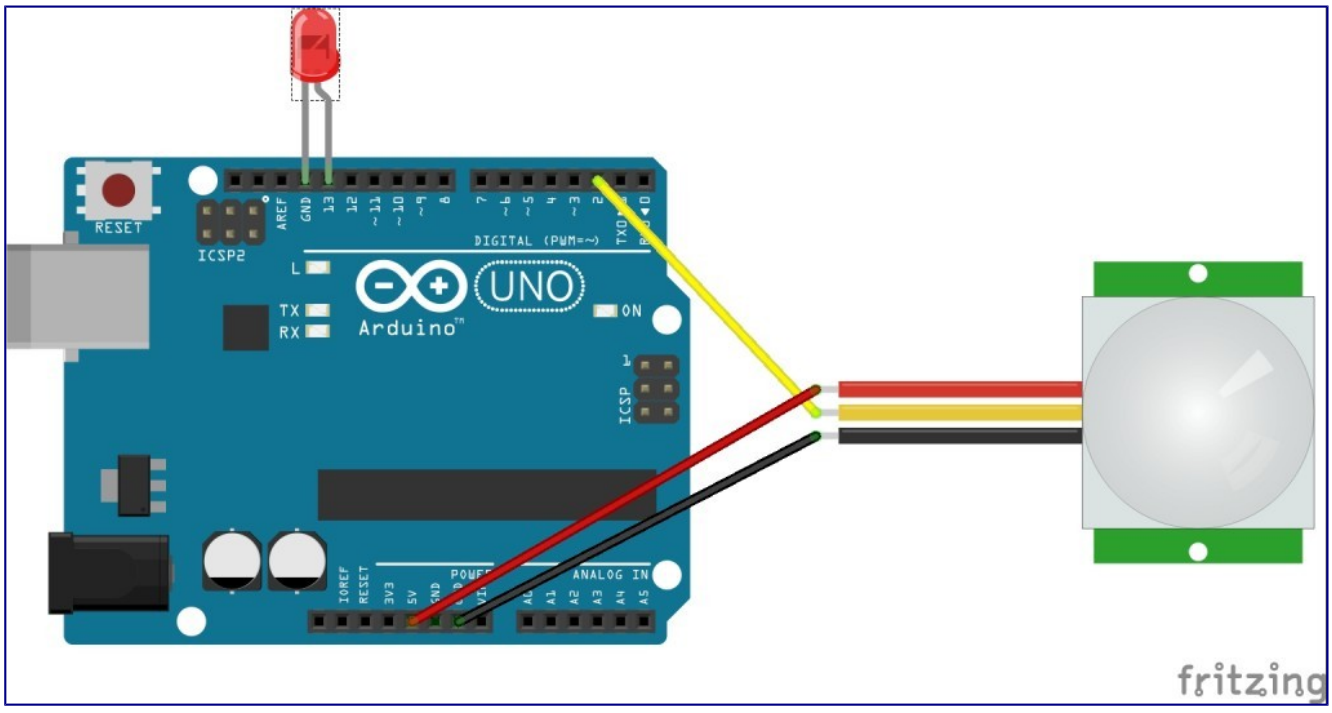


Les spécifications techniques sont les suivantes :

- Entrée : Courant continue de 4.5 à 20V
- Sortie : High 3.3 V / Low 0V (Détection ou non)
- Angle : <100 °
- Dimension : 32 mm * 24 mm
- Délai : de 5 à 200 secondes (ajustable)
- Portée : de 3 à 7 mètres (ajustable).
- Au repos : 50 microampères.

Le montage

Au vu des éléments nécessaires, vous allez voir, c'est plutôt simple !



Branchement du PIR sur l'arduino Uno

On branche donc :

- Le VCC du Pir sur le 5V de l'arduino
- Le GRD du PIR sur le GRD de l'arduino
- La dernière branche sur le pin 2 de l'arduino
- On ajoute une led de contrôle entre le pin 13 et un GRD de l'arduino

Mais attention : D'un PIR à l'autre, les branchements sont inversés ! J'avais 2 types de PIR, et j'ai grillé 2 PIR avant de me rendre compte que le branchement + et – était différent entre les 2 types de capteurs. BIEN SE RENSEIGNER AUPRÈS DU REVENDEUR !

Le code

Il est enfin temps de faire fonctionner tout ça ! Rien de bien compliqué non plus. On va commencer par laisser 30 secondes au PIR pour se calibrer, puis un fois cela fait, on va en boucle relever la valeur que nous renvoi le capteur : 0 ou 1. 0 signifiant pas de signal et 1 signifiant qu'il détecte une variation infrarouge.

Le code en lui même est disponible sur mon dépôt github : ici.

Si vous n'êtes pas à l'aise avec github, je vous le reproduis ci dessous :

```
//the time we give the sensor to calibrate (10-60 secs according to the
datasheet)
int calibrationTime = 30;

int ledPin = 13;                // choose the pin for the LED
int inputPin = 2;               // choose the input pin (for PIR sensor)
int pirState = LOW;             // we start, assuming no motion detected
int val = 0;                    // variable for reading the pin status

void setup() {
  pinMode(ledPin, OUTPUT);      // declare LED as output
  pinMode(inputPin, INPUT);     // declare sensor as input
  Serial.begin(9600);

  Serial.print("calibrating sensor ");
  for(int i = 0; i < calibrationTime; i++){
    Serial.print(".");
    delay(1000);
  }
}

void loop(){
  val = digitalRead(inputPin);  // read input value
  Serial.println(val);
  if (val == HIGH) { // check if the input is HIGH
    digitalWrite(ledPin, HIGH);  // turn LED ON
    delay(150);

    if (pirState == LOW) {
      // we have just turned on
      Serial.println("Motion detected!");
      // We only want to print on the output change, not state
      pirState = HIGH;
    }
  } else {
    digitalWrite(ledPin, LOW); // turn LED OFF
    delay(300);
    if (pirState == HIGH){
      // we have just turned of
      Serial.println("Motion ended!");
      // We only want to print on the output change, not state
      pirState = LOW;
    }
  }
```

```
}
```

On commence donc par initialiser les pins dans leur état d'entrée ou de sortie, puis on calibre le capteur.

Puis dans le loop, on relève la valeur renvoyée par le capteur et on agit en conséquence. Quand vous lancez le moniteur, vous avez alors un retour comme ceci :

```
calibrating sensor ..............................0
0
0
0
0
0
1
Motion detected!
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
0
Motion ended!
```

Ce que le moniteur nous ressort

Bonus : Créer une lampe qui s'allume automatiquement

On va profiter de ce capteur pour l'utiliser dans un cas pratique ! Nous allons le coupler avec un relai, qui va nous permettre de contrôler du courant (du 220V par exemple) pour allumer ou éteindre une lampe lorsque l'on détecte un mouvement. Cette deuxième partie est donc un cas pratique d'utilisation de notre capteur de mouvement.

Le matériel nécessaire, en plus est le suivant :

- Un relai
- Une lampe (n'importe quelle lampe pas trop chère fera l'affaire)
- Un domino électrique

Pour la lampe, j'ai acheté une lampe à LED chez Castorama à 12€. Vu qu'il faut couper le câble d'alimentation pour le montage, évitez de sacrifier une lampe de qualité (ça vaut mieux pour la paix des ménages).

Un relais, ou relais électromagnétique est, selon wikipedia un organe électrique permettant de dissocier la partie puissance de la partie commande, autrement dit, Il permet l'ouverture / la fermeture d'un circuit électrique par un second circuit totalement isolé. Ici, nous allons donc contrôler l'ouverture ou la fermeture du circuit en 220V (enfin 12V sur ma lampe de test) avec un second circuit en 5V contrôlé par la carte Arduino.

Voila à quoi va désormais ressembler notre montage :

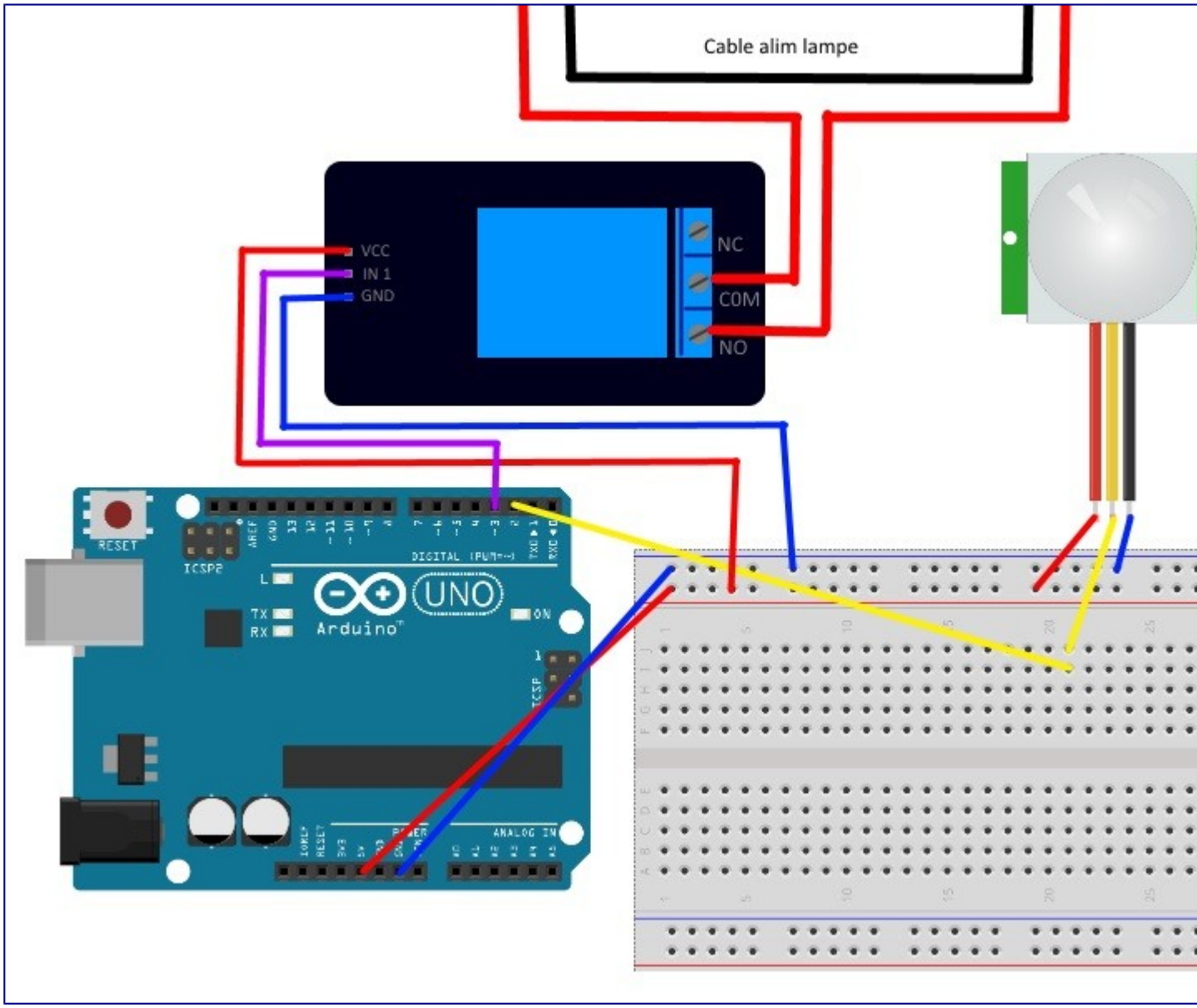
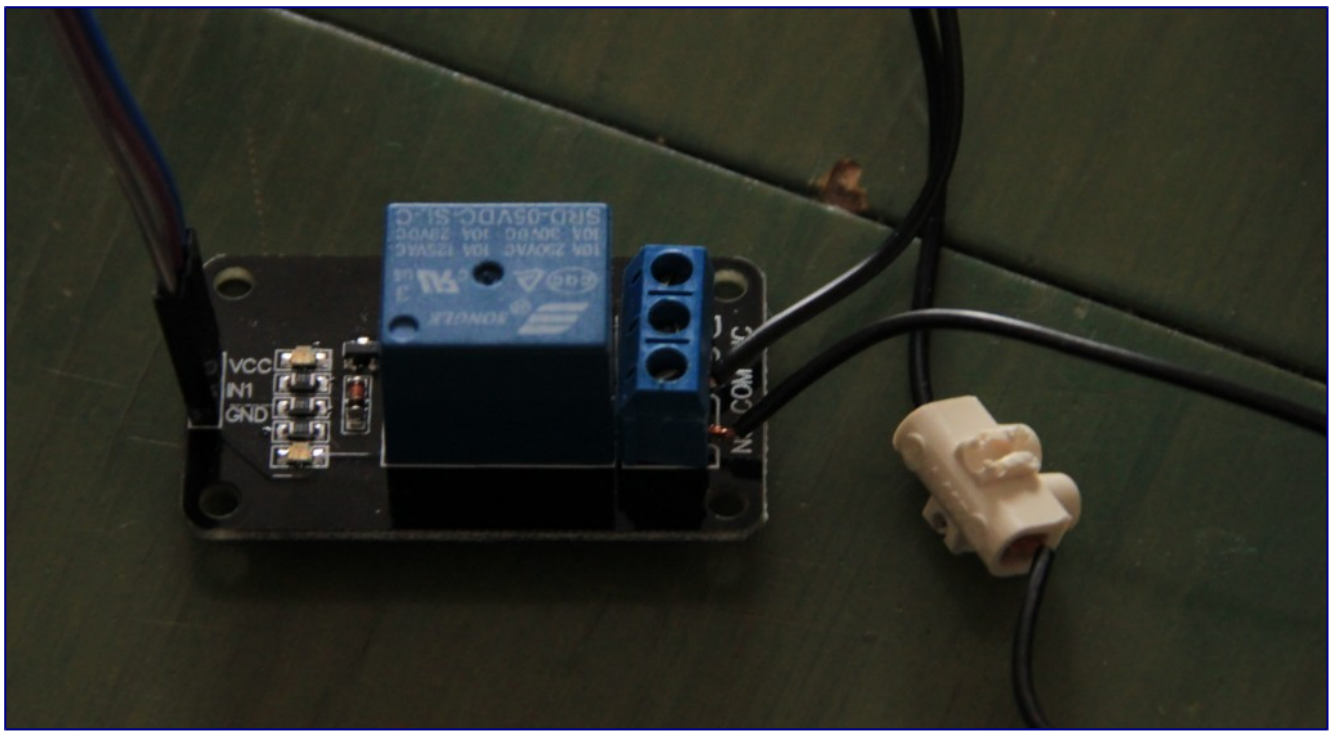


Schéma du montage PIR et relais

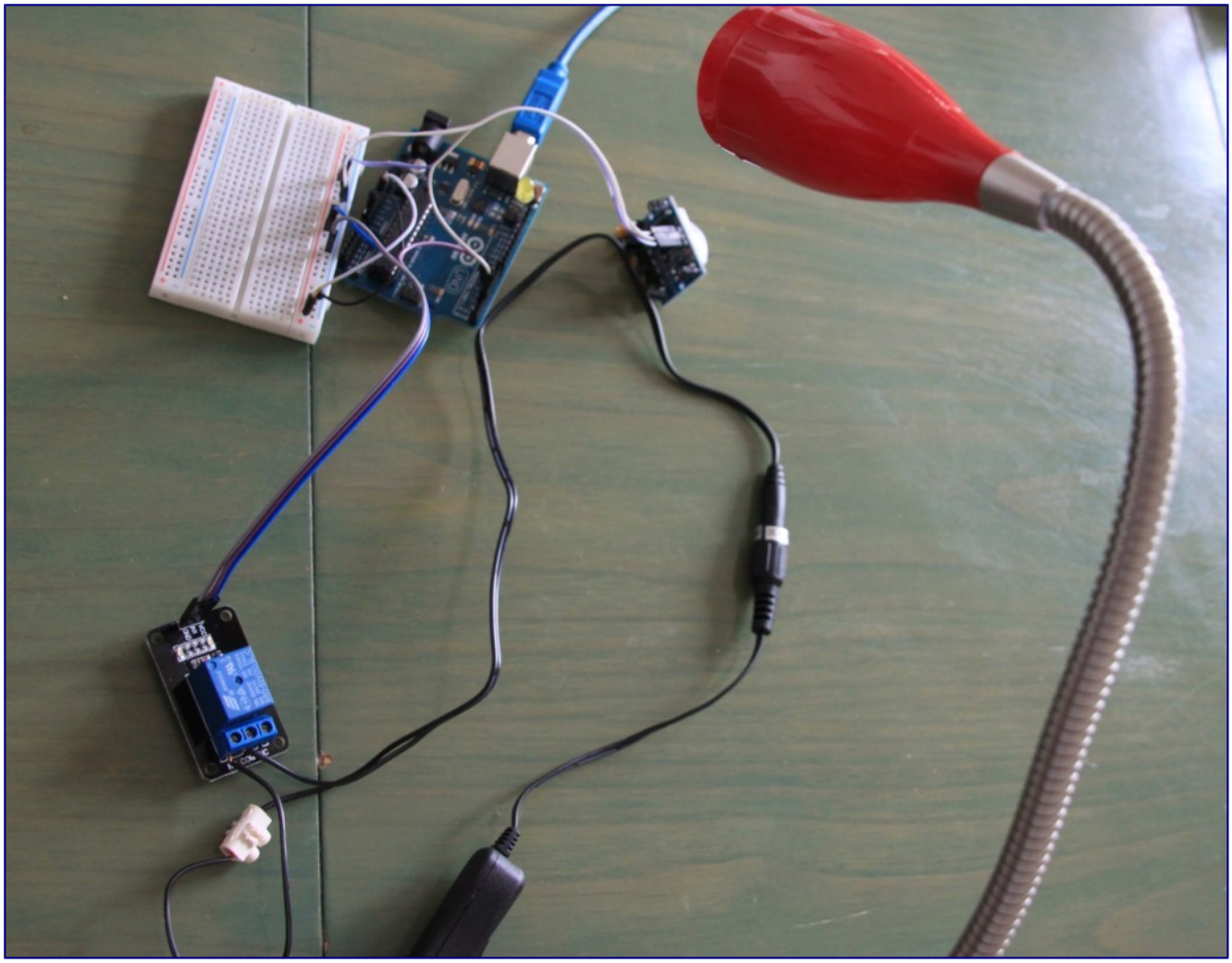
Comme vous pouvez le voir, le montage du PIR ne varie pas, on le connecte toujours aux mêmes bornes.

Pour le relais, il faut faire 2 choses : Connecter GND à la masse commune du montage (pir et relais doivent avoir la même masse), connecter le VCC au 5V fourni par l'arduino, et connecter le pin de données au port 3 de l'arduino.

De l'autre côté du relais, il va falloir couper votre câble d'alimentation de la lampe. On va donc avoir quelque chose qui ressemble à ça :



Contrôle de l’alimentation de la lampe grâce au module relais



Le montage dans son ensemble

Avant de vous donner le code, voici une petite vidéo qui vous montre comment tout cela fonctionne :

Passons au code désormais. Comme vous pouvez vous en douter, on va utiliser le code de la première partie du tutoriel que l'on va enrichir pour prendre en compte le contrôle du module relais.

Vous trouverez le code sur mon dépôt github, ici.

Si vous êtes githubophobe, vous trouverez le code ci dessous :

```
//the time we give the sensor to calibrate (10-60 secs according to the
datasheet)
int calibrationTime = 30;
// The time the device will stay on
int delayTime = 5000;

int ledPin = 13;                // choose the pin for the LED
int inputPin = 2;               // choose the input pin (for PIR sensor)
int relayPin = 3;
int pirState = LOW;             // we start, assuming no motion detected
int stateRelay = HIGH;
int val = 0;                    // variable for reading the pin status

void setup() {
  pinMode(ledPin, OUTPUT);      // declare LED as output
  pinMode(relayPin, OUTPUT);
  pinMode(inputPin, INPUT);     // declare sensor as input
  Serial.begin(9600);
  digitalWrite(relayPin, stateRelay);
  //give the sensor some time to calibrate
  Serial.print("calibrating sensor ");
  for(int i = 0; i < calibrationTime; i++){
    Serial.print(".");
    delay(1000);
  }
  Serial.println("SENSOR ACTIVE");
  delay(50);
}

void loop(){
  val = digitalRead(inputPin);  // read input value
  //Serial.println(val);
  if (val == HIGH) { // check if the input is HIGH
    digitalWrite(ledPin, HIGH);  // turn LED ON
    delay(150);

    if (pirState == LOW) {
      // we have just turned on
      Serial.println("Motion detected!");
      // We only want to print on the output change, not state
      pirState = HIGH;
    }
  } else {
    digitalWrite(ledPin, LOW); // turn LED OFF
    delay(300);
      if (pirState == HIGH){
      // we have just turned of
```

```
      Serial.println("Motion ended!");
      // We only want to print on the output change, not state
      pirState = LOW;
    }
  }

  Serial.println(pirState);
  if(pirState == HIGH){
     digitalWrite(relayPin, LOW);
     delay(delayTime);
  } else {
     digitalWrite(relayPin, HIGH);
  }
}
```

Comme vous pouvez le voir, rien de bien compliqué ! Comme dans le premier montage, on contrôle si on détecte un mouvement, et si oui, dans ce cas on va activer le module relais pour allumer la lampe.

Et voila, c'est fini pour ce tutoriel.

Précaution importante

Ce montage est assez simpliste et n'est en soit pas sécurisé pour l'arduino. En effet, le fait d'ouvrir ou fermer un relai va générer des surtensions. Si vous souhaitez utiliser ce montage de manière régulière, il va donc être IMPÉRATIF de protéger votre circuit. Heureusement, c'est assez simple.

Je vous invite à lire ce tuto pour en savoir plus et protéger votre circuit. L'idée est d'utiliser une diode pour protéger notre arduino des retours potentiels du relais. Il se peut aussi que votre module relais possède déjà cette sécurité, donc à vous de vérifier en fonction du module que vous avez acheté !

Voila, c'est enfin tout pour ce tutoriel. Si vous avez des questions, n'hésitez pas !

Source :
http://www.manuel-esteban.com/tuto-lire-un-capteur-infrarouge-avec-arduino/