fabrication, impression 3D, modelisation 3D, openscad, photographie, raspberry-pi, tube iro, em

Lightbox

(Page créée le 8 avril 2022)

Une boite d'éclairage pour photographier des photomontages

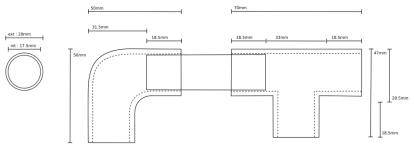
Structure en tube PVC IRO, styroglass translucide, raspberry pi avec camera

Structure

Le tube IRL (Isolant Rigide Lisse) ou tube IRO (Isolant Rigide Ordinaire) est utilisé dans le bâtiment comme gaine pour les câbles électriques. Il est en PVC, en général gris ou blanc et à l'avantage de fournir un matériau très économique pour construire des structures. On peut assembler les tubes avec différents types de raccords : manchons de raccordement, pièces en T, en L, coudes ou courbes. On trouve aussi des fixations de différents types pour les fixer dans un mur. Dimensions communes : 2,40 m de longueur pour un diamètre de 16mm, 20mm, 25mm, 32mm ou 40mm.

Dimensions utiles

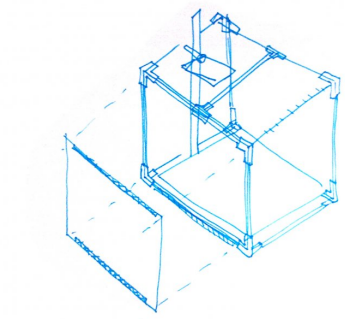
Quelques nombres bien utiles pour réaliser des structures avec des tubes IRO.



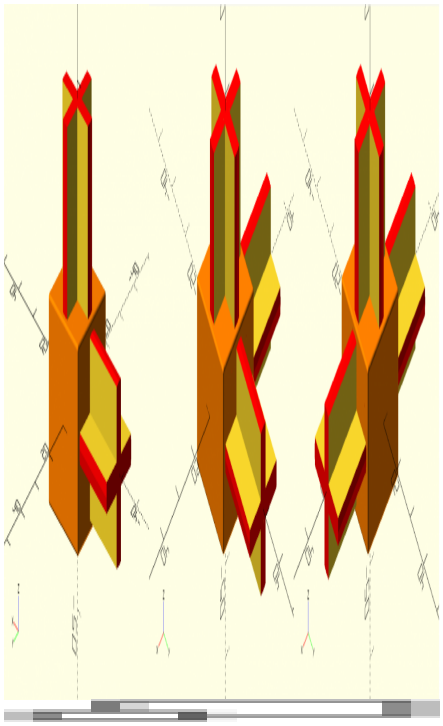
Projet

Premier croquis, une structure en tube IRO porte deux morceaux de styroglass translucide qui diffuseront la lumière projetée de l'extérieur. La structure en tubes IRO est assemblée à partir de pièces de raccord imprimées en 3D.

~~Les pièces de ce projet sont compatibles avec les coudes et pièces en té existantes. Mais elles sont pleines, donc on ne peut plus s'en servir comme gaine!~~ Finalement non, les pièces ne sont pas compatibles mais s'articulent autour d'un cube de 20mm.



Raccords IRO



3 types de raccord : en L, en L 3D, en T

Les 3 pièces nécessitent le module roundedcube de Daniel Upshaw.

iro_raccord_l.scad (cliquer pour afficher le code)

iro_raccord_l.scad

```
/*
    Raccords pour tube IRO
    Debian 9.5 Stretch @ kirin / 20220410 / pierre @ lesporteslogiques.net
    OpenSCAD 2021.1
      + roundedcube.scad (par Daniel Upshaw) : https://danielupshaw.com/openscad-rounded-corners/
*/


use <roundedcube.scad>;

color([1, 0.5, 0]) roundedcube(size = [20, 20, 20], center = true, radius = 0.5, apply_to = "all");
translate([0, 0, 18.5]) croix();
rotate([0, 90, 0]) translate([0, 0, -18.5]) croix();

module socle_cylindre() {
    color([0, 1, 0]) cylinder($fn=72, h=2.5, r=11.25, center=true);
}

module croix() { // hauteur totale 19.5, dont 1 à inclure dans lautre partie
    intersection() {
        color([1, 0, 0]) cylinder($fn=36, h=18.5+1, r=8.75, center=true);
        union() {
            cube(size=[17.5+0.3, 3, 18.5+1.3], center=true);
            rotate([0, 0, 90]) cube(size=[17.5+0.3, 3, 18.5+1.3], center=true);
        }
    }
}
```

iro_raccord_l3d.scad (cliquer pour afficher le code)

iro_raccord_l3d.scad

```
/*
    Raccords pour tube IRO
    Debian 9.5 Stretch @ kirin / 20220410 / pierre @ lesporteslogiques.net
    OpenSCAD 2021.1
      + roundedcube.scad (par Daniel Upshaw) : https://danielupshaw.com/openscad-rounded-corners/
*/


use <roundedcube.scad>;

color([1, 0.5, 0]) roundedcube(size = [20, 20, 20], center = true, radius = 0.5, apply_to = "all");
```

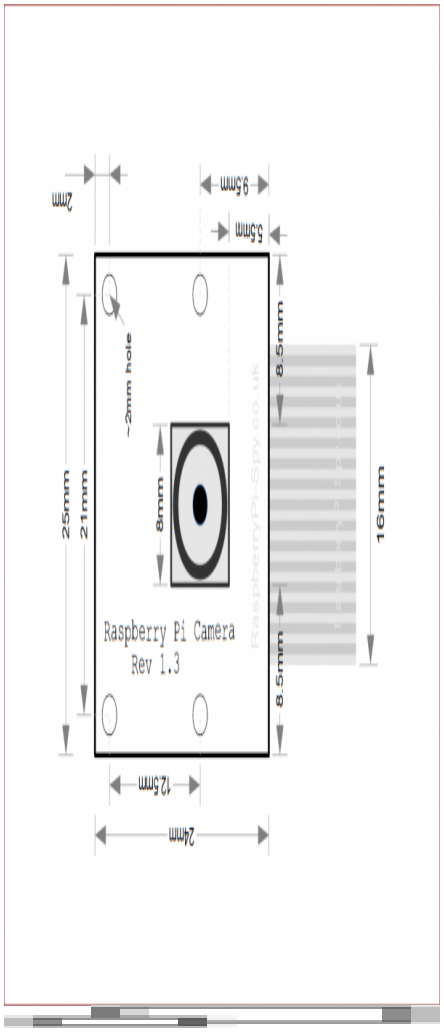
```
translate([0, 0, 18.5]) croix();
rotate([0, 90, 0]) translate([0, 0, -18.5]) croix();
rotate([90, 270, 0]) translate([0, 0, -18.5]) croix();

module socle_cylindre() {
    color([0, 1, 0]) cylinder($fn=72, h=2.5, r=11.25, center=true);
}

module croix() { // hauteur totale 19.5, dont 1 à inclure dans lautre partie
    intersection() {
        color([1, 0, 0]) cylinder($fn=36, h=18.5+1, r=8.75, center=true);
        union() {
            cube(size=[17.5+0.3, 3, 18.5+1.3], center=true);
            rotate([0, 0, 90]) cube(size=[17.5+0.3, 3, 18.5+1.3], center=true);
        }
    }
}
```

iro_raccord_t.scad (cliquer pour afficher le code)

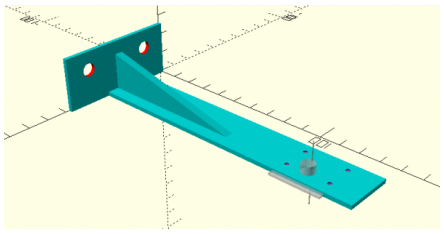
iro_raccord_t.scad

```
/*
    Raccords pour tube IRO
    Debian 9.5 Stretch @ kirin / 20220410 / pierre @ lesporteslogiques.net
    OpenSCAD 2021.1
      + roundedcube.scad (par Daniel Upshaw) : https://danielupshaw.com/openscad-rounded-corners/
*/


use <roundedcube.scad>;

color([1, 0.5, 0]) roundedcube(size = [20, 20, 20], center = true, radius = 0.5, apply_to = "all");
translate([0, 0, 18.5]) croix();
rotate([0, 90, 0]) translate([0, 0, -18.5]) croix();
rotate([0, 270, 0]) translate([0, 0, -18.5]) croix();

module socle_cylindre() {
    color([0, 1, 0]) cylinder($fn=72, h=2.5, r=11.25, center=true);
}

module croix() { // hauteur totale 19.5, dont 1 à inclure dans lautre partie
    intersection() {
        color([1, 0, 0]) cylinder($fn=36, h=18.5+1, r=8.75, center=true);
        union() {
            cube(size=[17.5+0.3, 3, 18.5+1.3], center=true);
            rotate([0, 0, 90]) cube(size=[17.5+0.3, 3, 18.5+1.3], center=true);
        }
    }
}
```

Pièce de soutien pour la camera



Dimensions du module de caméra Raspberry Pi (source raspberrypi-spy.co.uk)



plateforme_picam_2.scad (cliquer pour afficher le code)

plateforme_picam_2.scad

```
/*
    Support Picam
    Debian 9.5 Stretch @ kirin / 20220411 / pierre @ lesporteslogiques.net
    OpenSCAD 2021.1
*/

difference() {
    color([0, 0.8, 0.8]) cube(size=[58, 2, 26], center=false);
    color([1, 0, 0]) percement_fixation();
}
difference() {
    color([0, 0.8, 0.8]) translate([12, 0, 0])cube(size=[24, 140, 2], center=false);    // plateforme
    translate([27, 115, 1]) rotate([0, 0, 90]) percement_picam();  // percement_picam
}


color([0, 0.8, 0.8]) translate([30, 0, 0]) rotate([270, 0, 90]) renfort(4);
//color([0, 0.8, 0.8]) translate([58, 0, 0]) rotate([270, 0, 90]) renfort();


module renfort(ep = 2) {
    linear_extrude(ep) {
        polygon(
                points=[ [0,0], [0,-20] ,[80,0] ],
```

```
                paths =[ [0,1,2] ]
            );
        };
    }

    module percement_fixation() {
        translate([12, 2, 17]) rotate([0, 90, 90])
            cylinder($fn=36, h=6, r=3.2, center=true);
        translate([46, 2, 17]) rotate([0, 90, 90])
            cylinder($fn=36, h=6, r=3.2, center=true);
    }

    module percement_picam(startx = 0, starty = 0, startz = 0) {
        {
            % translate([startx, starty, startz]) %cube([25, 24, 1], center = true);   // forme pcb picam
            % translate([startx, starty, startz-15]) cylinder($fn = 3, h = 30, r=0.3); // axe
            % translate([startx, starty -2.5, 4]) cylinder($fn=36, h=4, r=3.2, center=true); // objectif picam
        }
        color([0.5, 0, 0.8]) translate([startx - 10.5, starty -2.5, startz]) cylinder($fn=36, h=4, r=1, center=true);
        color([0.5, 0, 0.8]) translate([startx + 10.5, starty -2.5, startz]) cylinder($fn=36, h=4, r=1, center=true);
        color([0.5, 0, 0.8]) translate([startx - 10.5, starty +10, startz]) cylinder($fn=36, h=4, r=1, center=true);
        color([0.5, 0, 0.8]) translate([startx + 10.5, starty +10, startz]) cylinder($fn=36, h=4, r=1, center=true);
    }
```

→ trop souple, nécessite plsu de renforts pour rester rigide

Raspberry Pi + camera

Modèle de la caméra : **Raspberry Pi camera Rev. 1.3** (definition 2592 x 1944)
Modèle Rpi : **Raspberry Pi 2 Model B v1.2** (Citron)

Pour avoir des infos complètes sur le modèle de Rpi `cat /proc/cpuinfo`, la ligne «Revision» donne la version qu'on peut retrouver dans cette liste (dans le cas présent : Revision a22042, 2 Model B (with BCM2837))

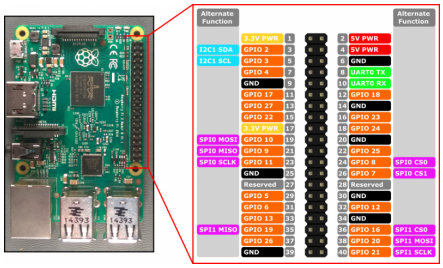
Brancher et configurer la camera (source) :

- Brancher la caméra sur le rpi
- Configurer dans les préférences (redémarrer si nécessaire)
- Vérifier que la caméra est reconnue : `vcgencmd get_camera`
- Tester une capture d'image en ligne de commande `raspistill -o Desktop/image.jpg`
- Pour une prévisualisation : `raspistill -k` (Ctrl+C pour stopper)

A ce stade, beaucoup de questions :

- Quelle définition adaptée à l'imprimante à sublimation (2560 x 1710 pour l'imprimante) ?
- Quels réglages peut-on faire sur l'image ?
- Comment configurer un point d'accès wifi pour récupérer les images ?
- Est ce possible de relier directement à l'imprimante à sublimation (= sans passer par carte SD) ?
- Quelle version de python3 ? `python3 –version` → python 3.5.3
- Quelle version de picamera ? `pip3 freeze` → picamera==1.13

Ajouter un bouton



Le bouton est relié à GND (broche 6) à droite, et pour le signal à GPIO17 (broche 11)

Utilisation directe de l'imprimante à sublimation

Comment imprimer directement sur l'imprimante Canon Selphy CP730 ?

`hostnamectl # renvoie la version du système`

→ Raspbian GNU/Linux 9 (stretch)

```
sudo apt update          # mise à jour
sudo apt install cups    # installation
sudo apt install gcc libtool libssl-dev libc-dev libjpeg62-turbo-dev libpng-dev libtiff5-dev
sudo apt install libcups2-dev
sudo apt install printer-driver-gutenprint
```

télécharger gutenprint :
https://sourceforge.net/projects/gimp-print/files/gutenprint-5.3/5.3.3/gutenprint-5.3.3.tar.xz/download
extraire et se rendre dans le dossier, puis

```
./configure
make clean
make
sudo make install
```

`sudo usermod -a -G lpadmin pi # ajouter l'utilisateur pi au groupe lpadmin`

`sudo service cups restart # redémarrer cups`

En cours

https://elinux.org/R-Pi_Troubleshooting

https://raspberry-lab.fr/Composants/Utilisation-Camera-sur-Raspberry-Pi-Francais/

https://raspberrytips.fr/camera-raspberry-pi-comme-webcam/

https://projects.raspberrypi.org/en/projects/push-button-stop-motion/2

https://stackoverflow.com/questions/25609844/pi-camera-preview-with-gui-raspberry-pi

https://pyimagesearch.com/2016/08/29/common-errors-using-the-raspberry-pi-camera-module/

Code

Comment faire quand la preview reste présente à l'écran alors que le programme est terminé ?
→ ouvrir un terminal avec CTRL+ALT+T et taper en aveugle `pkill python3`

En cours !

```
#coding: UTF-8
from picamera import PiCamera
from time import sleep
import datetime

# camera = PiCamera()
# https://picamera.readthedocs.io/en/release-1.13/fov.html?highlight=sensor%20mode#sensor-modes
# camera = PiCamera(resolution=(2592, 1944), framerate=15,sensor_mode=2)
camera = PiCamera()
# camera.hflip = True
# camera.vflip = True
camera.rotation = 180
camera.resolution = (2592, 1944)
camera.framerate = 15            # necessaire pour changer la resolution
camera.preview_fullscreen=False
camera.preview_window=(620, 120, 648, 486)


camera.start_preview()
#camera.brightness = 50           # de 0 à 100
#camera.contrast = 0              # de -100 à 100
"""
image_effect : none, negative, solarize, sketch, denoise, emboss,
  oilpaint, hatch, gpen, pastel, watercolor, film, blur, saturation,
  colorswap, washedout, posterise, colorpoint, colorbalance, cartoon,
  deinterlace1, deinterlace2
"""
#camera.image_effect = 'none'
"""
exposure_mode : off, auto, night, nightpreview, backlight, spotlight
  sports, snow, beach, verylong, fixedfps, antishake, fireworks
"""
```

```
#camera.exposure_mode = 'auto'
"""
awb_mode (balance des blancs) : off, auto, sunlight, cloudy, shade,
  tungsten, fluorescent, incandescent, flash, horizon
"""
camera.awb_mode = 'fluorescent'
camera.annotate_text = "awb_mode: fluorescent"


#~ for awb in camera.AWB_MODES:
    #~ camera.awb_mode = awb
    #~ camera.annotate_text = "awb_mode: %s" % awb
    #~ sleep(3)
sleep(5)
# camera.capture('/home/pi/Desktop/image_test_1.jpg')
# jpg ou png indiqué par l'extension
camera.capture('/home/pi/Desktop/picam_' + str(datetime.datetime.now()) + '.png')
camera.stop_preview()
```

\\

Ressources

Tout sur les caméras Raspberry Pi : https://www.raspberrypi.com/documentation/accessories/camera.html
Démarrer avec la caméra : https://projects.raspberrypi.org/fr-FR/projects/getting-started-with-picamera/0
Documentation **picamera** : https://picamera.readthedocs.io/en/release-1.13/
Utiliser les **GPIO** avec la lib. RPi.GPIO : https://roboticsbackend.com/raspberry-pi-gpio-interrupts-tutorial/
Doc GPIO Zero : https://gpiozero.readthedocs.io/en/stable/
Quelques exemples **GPIO Zero** : https://www.framboiseetcompagnie.fr/1220-2/
Pour un rendu cohérent des images (luminosité, constraste, etc.), voir
https://picamera.readthedocs.io/en/release-1.13/recipes1.html?highlight=capture#capturing-consistent-images
overlay avec imagemagick :
https://raspi.tv/2014/overlaying-text-and-graphics-on-a-photo-and-tweeting-it-pt-5-twitter-app-series

Article extrait de : http://lesporteslogiques.net/wiki/ - **WIKI Les Portes Logiques**
Adresse : **http://lesporteslogiques.net/wiki/openatelier/projet/lightbox?rev=1649889758**
Article mis à jour: **2022/04/14 00:42**