# Scanner 3D Volant de Bâtiments

## Slave Application

GUO Kai - ZHANG Zhengmeng - SUN Xuan

# Sommaire

- Contexte
- Solution
- Développement
- Architecture
- Travail Réalisé
- Conlusion

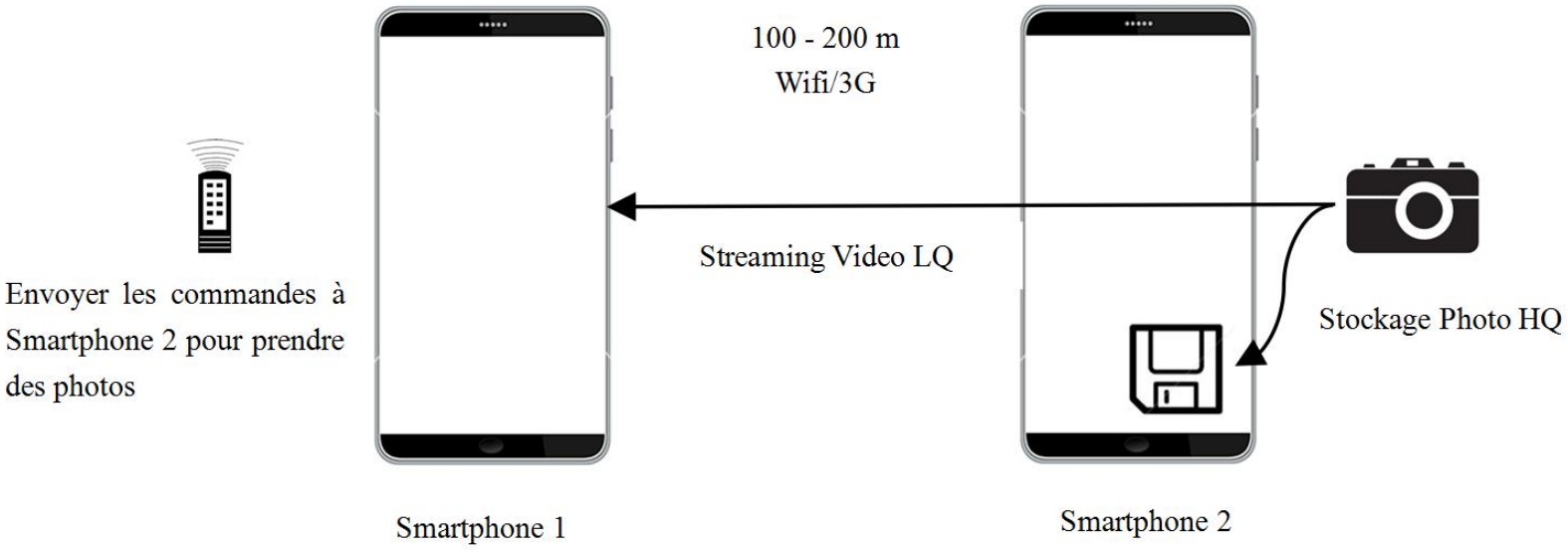
# Contexte

Ce projet a pour objectif de réaliser un scanner 3D automatique de bâtiments.

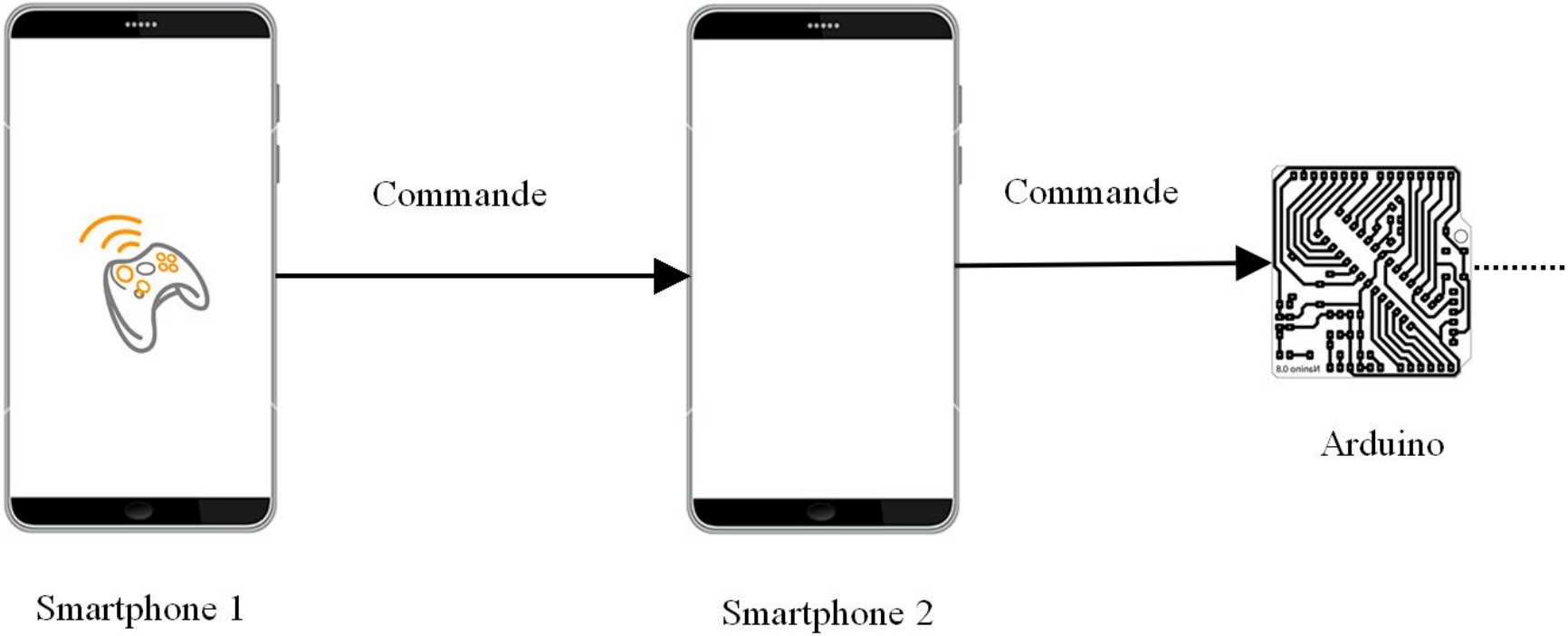
Le principe est le suivant :

Un drone volant acquière des images HQ des différentes faces et toits d'un bâtiment. Les images prises servent à reconstruire une image 3D du bâtiment.

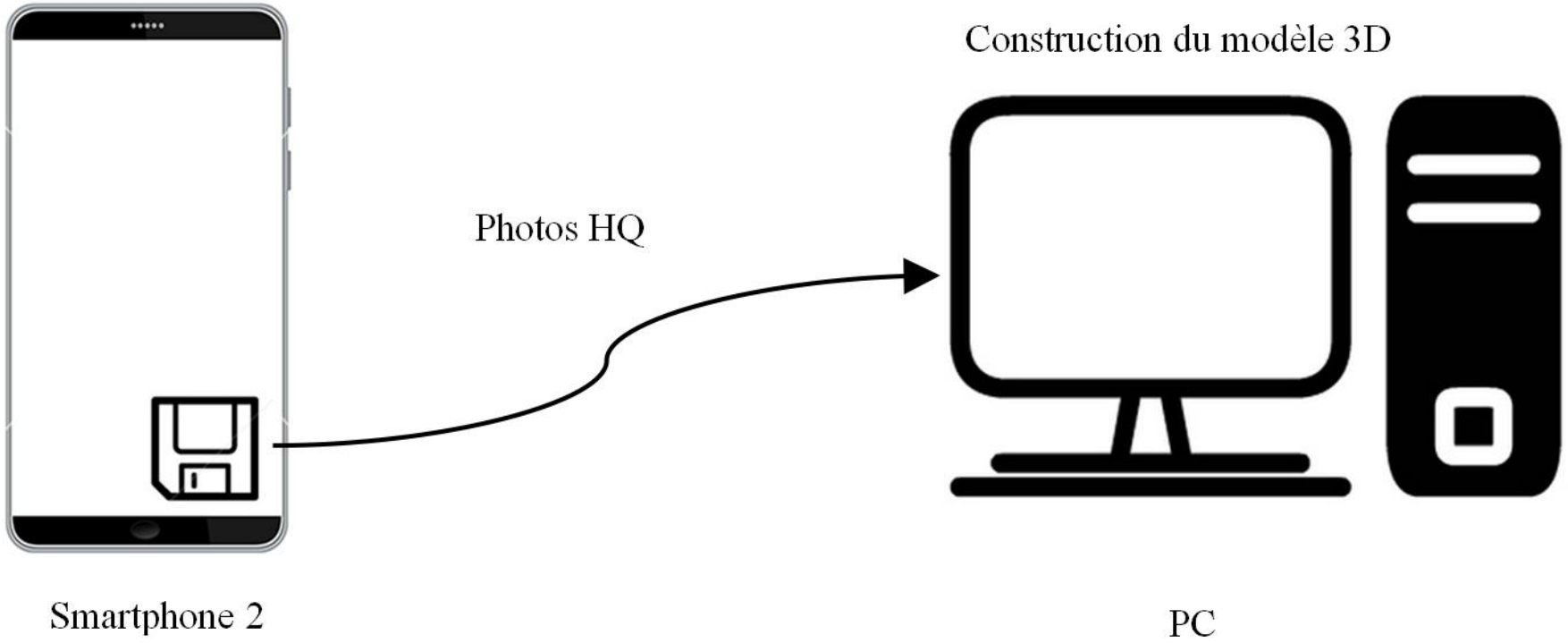
# Solution (étape 1/3)



# Solution (étape 2/3)



# Solution (étape 3/3)

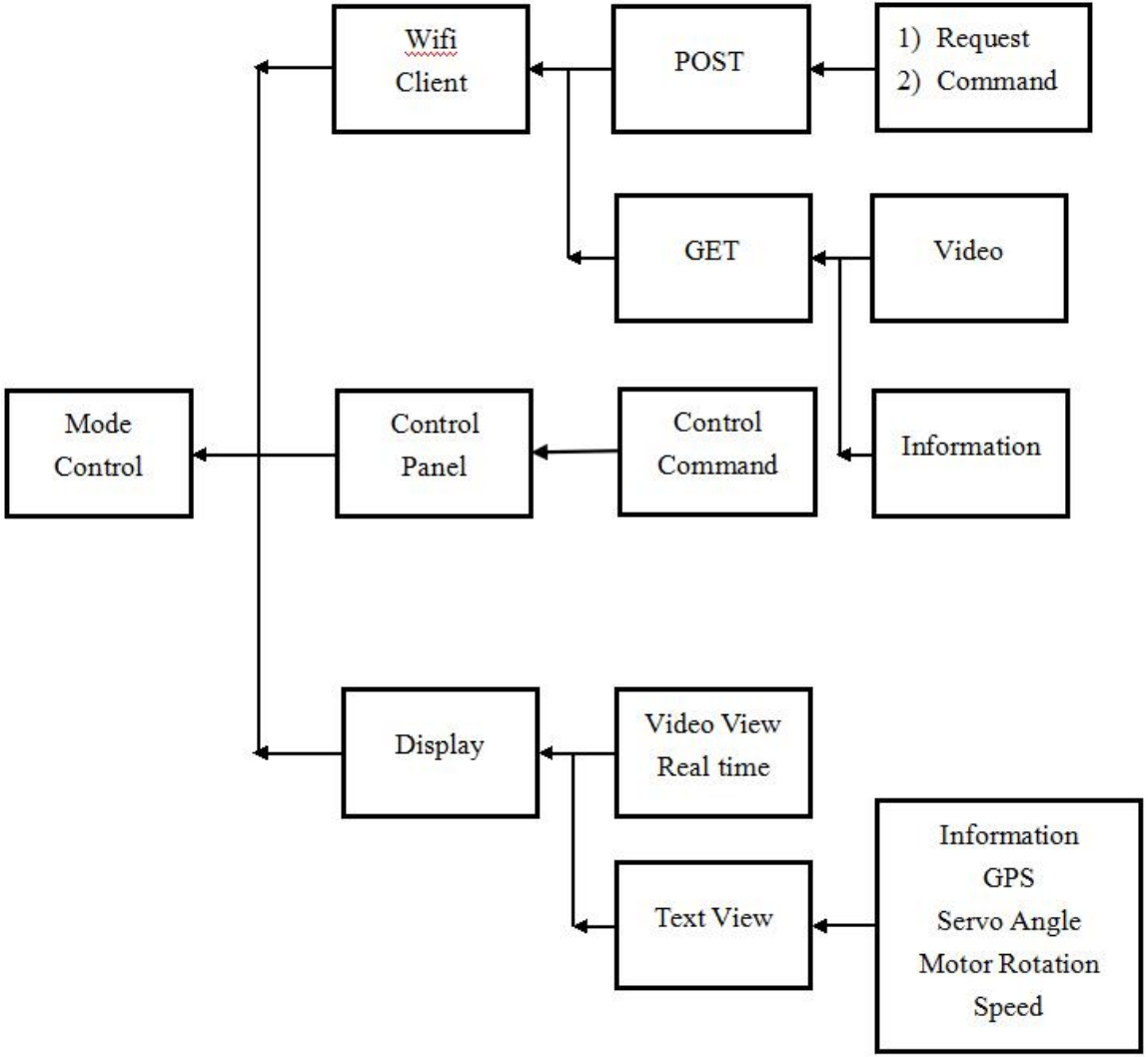


# Développement

arduino uno
polulu 6pin
intel AZ210b(android 2.3.7)
adt eclipse
Visual Studio
OpenCV
OpenGL
lib SDL
POSIX threads

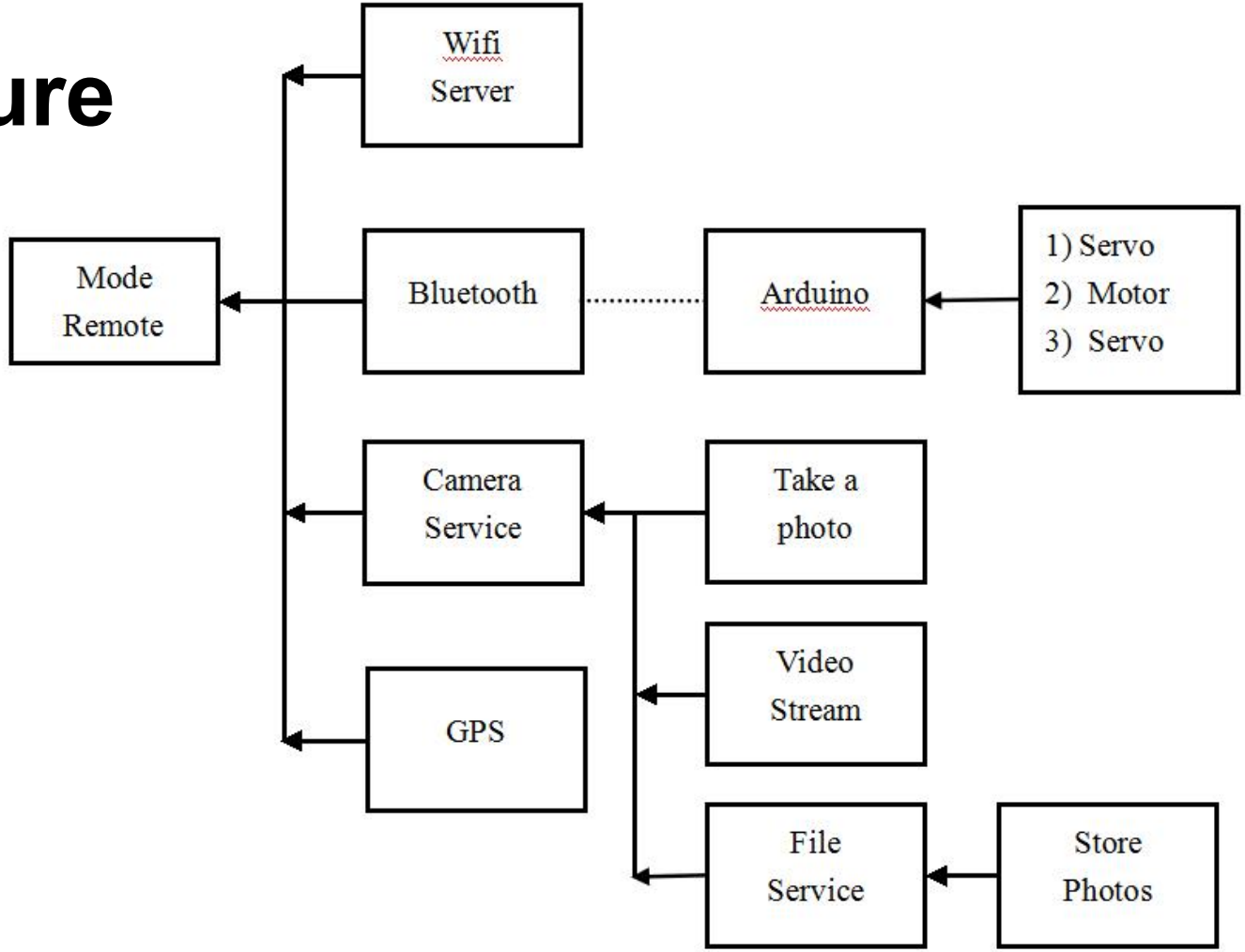
# Architecture

Control Mode



# Architecture

Remote Mode



# Travail Réalisé

## Mode Remote:

Bluetooth(smartphone 2--arduino)

Camera

GPS

## Mode Command:

Control Panel

Display

# Camera

```
private void takePicOnPicMode() {
    // TODO Auto-generated method stub
    iCamera.takePicture(null, null, new PictureCallback() {

        @Override
        public void onPictureTaken(byte[] data, Camera camera) {
            // TODO Auto-generated method stub
            try {
                Bundle tempData = new Bundle();
                tempData.putByteArray("byte", data);//save picture binary data
                savePhoto(data);//save pic data into SD card
                Toast.makeText(getApplicationContext(), "Success", Toast.LENGTH_SHORT).show();
            } catch (Exception e) {
                // TODO: handle exception
                e.printStackTrace();
            }
        }
    });
```

# Camera

```
private File savePhoto(byte[] data){
    // TODO Auto-generated method stub
    File fileFolder = new File(pic_Dir);
    if (!fileFolder.exists()) {//if this dir doesn't exist  creat
        fileFolder.mkdir();
    }
    File picFile = new File(fileFolder, get_Pic_Name());

    try {
        fileOutputStream = new FileOutputStream(picFile);
        fileOutputStream.write(data);
        fileOutputStream.close();
    } catch (FileNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
        return null;
    }
    return picFile;

}
```

# GPS

```
private void startGPS() {
    // TODO Auto-generated method stub
    if (!GPS_State) {
        // TODO Auto-generated method stub
        /*
         * launch the GPS service and display in the gps tag
         */
        myGpsLocationListener.onProviderEnabled(LocationManager.GPS_PROVIDER
        launceGPS(myGpsLocationListener);
        GPS_State = true;
        Remote_Terminal_GPS.setText(R.string.Remote_State_GPS_ON);
        Remote_Terminal_GPS.setTextColor(android.graphics.Color.GREEN);
    }
}

private void stopGPS() {
    // TODO Auto-generated method stub
    myGpsLocationListener.onProviderDisabled(LocationManager.GPS_PROVIDER);
    //myLocationManager.removeUpdates(myGpsLocationListener);
    Remote_Terminal_GPS.setText(R.string.Remote_State_GPS_OFF);
    Remote_Terminal_GPS.setTextColor(android.graphics.Color.RED);
    Remote_GPS_info.setText(R.string.Remote_GPS_info_TURN_OFF);
    GPS_State = false;
}
```

# Arduino

```
#include <SoftwareSerial.h>
#include <Servo.h>

//definition of servo
Servo servo_1;
Servo servo_2;
//Servo servo_3;
SoftwareSerial btSerial(10, 11); // RX, TX
char buf[128];//receiving and sending buffer

void setup(){
  servo_1.attach(3);//attaching the servo_1 on pin 3
  servo_2.attach(6);
  //servo_3.attach(9);

  Serial.begin(9600);
  Serial.println("ready!");
  btSerial.begin(9600);

`
```

# Opération Synchrone

```
public boolean onTouch(View v, MotionEvent event) {
    // TODO Auto-generated method stub
    switch (v.getId()) {
    case R.id.Control_Left:
        if (!URGENTINTERRUPT) {
            try {
                onTouchSemaphore.acquire();
                while (event.getAction() == MotionEvent.ACTION_DOWN) {
                    sendHigh(R.id.Control_Left);
                }
                onTouchSemaphore.release();
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }else {
            resetALL();
            break;
        }
        break;
```

# Control Panel

```
@Override
public void onClick(View v) {
    // TODO Auto-generated method stub
    switch (v.getId()) {
    case R.id.btn_play:
        startListning();
        startStreaming();
        break;
    case R.id.btn_capture:
        catchSnapShot();
        break;
    case R.id.reset:
        URGENTINTERRUPT = true;
        break;
    default:
        break;
    }
}
```

# Opération des fichiers

```
private void clearResource() {


/*
 * build a directory in device intern memory and put in some app
 */
private void buildResource() {

/*
 * copy files from a specified directory
 */
private void copyResourceFile(int rid, String targetFile)
```

# Création des modèles 3D

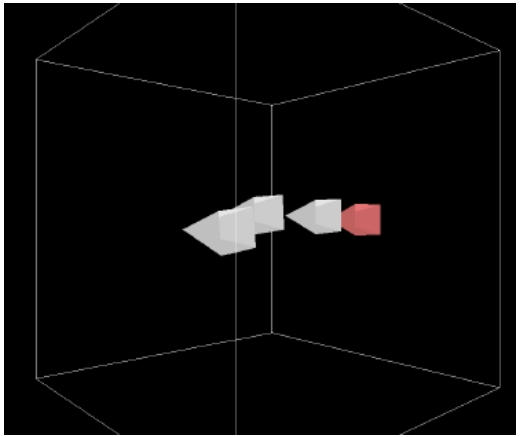
Traitement des photos obtenus

- Recherche les points-clés avec algorithme SIFT
- Faire correspondre contre des points-clés de la voisine
- Optimisation

# Création des modèles 3D

Traitement des photos obtenus

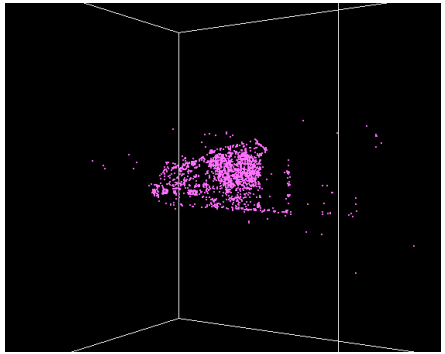
- Normalisation et étalonnage des paires d'images
- Obtenir les positions positions, les orientations et les paramètres des caméras

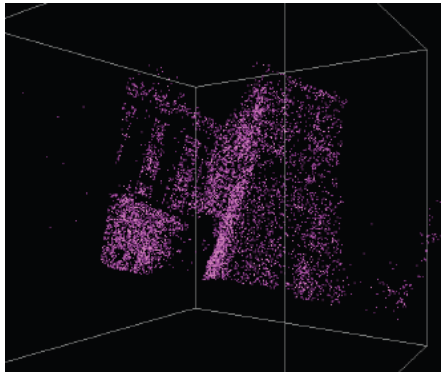


# Création des modèles 3D

Construire le nuage des points 3D

- Triangulation avec algorithm Delaunay
- Obtenir l’angle nécessaire pour construire les modèles en 3D

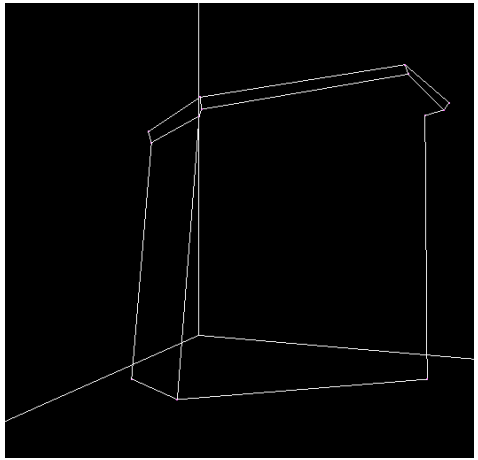


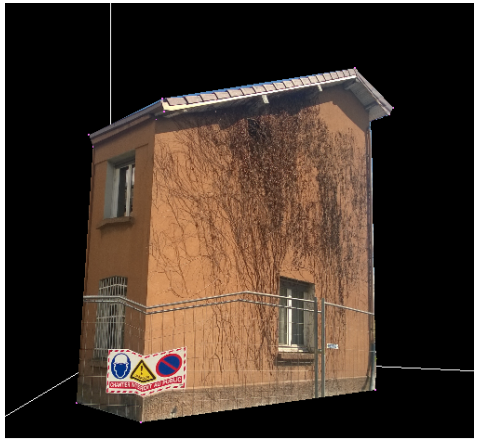


# Création des modèles 3D

Modélisation

- Construire manuellement du polygone
- Générer des textures





# Problèmes rencontrés

- Développement Android
- Streaming en temps réel
- Codage de data

# Fonctionalités à réaliser

- Network
- Test
- Intégration des fonctionnalités

# Conclusion

Tout d'abord nous approfondi largement nos connaissance en traitement d'image et développement Android.

Ensuite, nous avons découvert comment relier une partie logicielle avec une partie matérielle.