TP - Calibrage de caméra

Pour réaliser le TP, lancez la commande suivante :
source /tsi/tp/bin/tp-calib

1 RANSAC

L'objectif est de programmer un algorithme de détection de plans dans un nuage de points 3D en utilisant le principe de RANSAC.

1. Le script *ransacPlan.py* contient une ébauche de programme qui va vous permettre de débuter. A vous de le modifier pour qu'il effectue la détection de plans.
2. Pour visualiser les nuages de points, vous pouver tenter les commandes :
   /tsi/tp/bin/Linux/x86_64/asc2pn data.asc data.pn
   /tsi/tp/bin/Linux/x86_64/pmini data.pn
3. Une fois votre outil mis en place, vous pourrez tester sa robustesse par rapport aux deux paramètres : erreur permise et nombre d'itérations.
4. Vous pourrez tester également votre algorithme sur plusieurs jeux de données (les fichiers *.dat dans votre répertoire de travail).
5. Comment procéderiez-vous pour détecter plusieurs plans avec l'algorithme RANSAC (ne pas mettre en oeuvre) ?
6. Modifiez l'algorithme pour détecter deux plans parallèles.

2 Homographie

L'objectif est de mettre en place une procédure permettant de corriger l'effet de perspective. Les images que nous utiliserons seront des images d'un plan dans l'espace 3D ou des images acquises en rotation autour du centre optique.

2.1 Feuille A4

A partir du script *redres.py*, il s'agit de redresser l'image d'une page A4 afin de corriger l'effet de perspective : modèle de défomation, ajustement du modèle, modification des paramètres, robustesse aux points aberrants, temps de calcul, ...

Justifiez vos choix.

2.2 Panorama

Décrire la méthode mise en place pour faire la superposition de deux images dans le script *pano2.py*.

Quelle stratégie adopter pour réaliser un panorama avec ce script et les 8 images (001.jpg, .., 008.jpg) ?

2.3 Perspective

Corrigez la perspective de l'image de façade : *facade.tif*. Quelles informations vous manque-t-il pour le faire correctement ?