The **PUT RGB-D database** is publicly available for non-commercial, research purposes. By downloading the database researcher(s) agrees to the following restrictions:

1. The database cannot be further distributed, copied or published in any way. This includes sharing the database with other units or faculties in the requesting university/company.

2. All submitted papers or any publicly available text using the **PUT RGB-D database** must cite the following paper:

" An Indoor RGB-D Dataset for the Evaluation of Robot Navigation Algorithms " – A. Schmidt, M. Fularz, M. Kraft, A. Kasiński, M. Nowicki, Advanced Concepts for Intelligent Vision Systems, Series Lecture Notes in Computer Science, Volume 8192, 978-3-319-02894-1, 2013.

Bibtex citation:

```
@incollection{
      year={2013},
      isbn={978-3-319-02894-1},
      booktitle={Advanced Concepts for Intelligent Vision Systems},
      volume={8192},
      series={Lecture Notes in Computer Science},
      editor={Blanc-Talon, Jacques and Kasinski, Andrzej and Philips,
      Wilfried and Popescu, Dan and Scheunders, Paul},
      doi={10.1007/978-3-319-02895-8_29},
      title={An Indoor RGB-D Dataset for the Evaluation of Robot Navigation
      Algorithms},
      url={http://dx.doi.org/10.1007/978-3-319-02895-8_29},
      publisher={Springer International Publishing},
      keywords={robot navigation; SLAM; benchmark dataset; Kinect; RGB-D},
      author={Schmidt, Adam and Fularz, Michał and Kraft, Marek and
      Kasiński, Andrzej and Nowicki, Michał},
      pages={321-329}
}
```

3. All submitted papers or any publicly available text using the **PUT RGB-D database** will be reported to the owners of the database and listed in the "Publications" section of this page.

4. The institution represented by the researcher(s) will be listed in the "Institutions" section of this page.

Contact vision@put.poznan.pl for any information regarding the database.

PUT RGB-D database documentation

The database contains following folders and files:

- ***trajectory_X*** folder, which contains information about trajectory number X:
  - ***trajectory_X_kinect***
    - ***depth*** – folder which contains depth images registered by the Kinect sensor,
    - ***rgb*** – folder which contains colour (RGB) images registered by the Kinect sensor,
  - ***trajectory_X_robot_cameras***
    - ***robot_camera_left*** – folder which contains colour (RGB) images registered by the left camera,
    - ***robot_camera_right*** – folder which contains colour (RGB) images registered by the right camera,
  - ***trajectory_X_kinect_sample.avi*** – file which presents the data registered by Kinect RGB stream as a movie,
  - ***trajectory_X_robot_orientation_position.txt*** – file which stores ground-truth information about the robot orientation and position during the experiment
  - ***trajectory_X_Xsens.data*** – text file which stores the data registered from Xsens MTi sensor.
- ***camera_data.txt*** file in which camera parameters are stored.

Each ***trajectory_X_robot_orieintation_position.txt*** has the following structure:

- each line represents one robot position (values are separated with the semicolon ";" sign):
  - first value is the frame number,
  - next three values denotes rotation in Rodrigues notation with regard to the centre of the coordinate system,
  - last three values represents the position of the marker in the global coordinate system (x, y and z coordinates).

Each ***trajectory_X_Xsens.data*** file has the following structure:

- each line represents one measurement in the following format:
  tempearture | acc_x acc_y acc_z | quaternion_0 quaternion_1 quaternion_2 quaternion_3 | gyro_x | gyro_y | gyro_z<CR><LF>

File ***camera_data.txt*** contains all the parameters associated with the database:

- lines 2, 3 and 4 contains cameras parameters (cameras were calibrated using methods provided in OpenCV library):
  - fx, fu denotes focal lengths;
  - u0, v0 denotes principal points;
  - k1, k2, k3, k4, k5, k6 denotes radial distortion coefficients;
  - p1, p2 denotes tangential distortion coefficients,

- lines 7, 8, and 9 contains position and rotation with regard to the centre of calibration marker (centre of the robot coordination system). First three numerical values denotes rotation in Rodrigues notation, while other three denotes the translation in axis x, y and z axis.

Additional information:

- the quick preview of the registered sequence is provided on the following pages:
  http://youtu.be/XIJvHiWisAc
  http://youtu.be/feBJtGgVCZ0
  http://youtu.be/0iQ4BSzE9MU
  http://youtu.be/B3K_G_Sg5-Q
- the provided *.avi files are encoded with x264 (H264) codec by ffmpeg tool (tested in VLC media player),
- to get the full depth resolution images have to be loaded as 16-bit. Sample code snippet for this operation (using OpenCV):

```
cv::Mat depthImage = cv::imread(filename, CV_LOAD_ANY_DEPTH);
```

- the values in the depth image are given in millimetres,
- sample code snippet for undistorting the images from Kinect (using OpenCV):

```
// focal lengths
#DEFINE fx 517.3
#DEFINE fy 516.5

// principal point
#DEFINE cx 318.6
#DEFINE cy 255.3

// Kinect image
#DEFINE HEIGHT 480
#DEFINE WIDTH 640

// Distortion coefficients
#DEFINE k1 0.2624
#DEFINE k2 -0.9531
#DEFINE p1 -0.0054
#DEFINE p2 0.0026
#DEFINE k3 1.1633

float coeffs[5] = {k1, k2, p1, p2, k3 };
cv::Mat distCoeffs = cv::Mat(1, 5, CV_32FC1, &coeffs);

float cm[3][3] =  {{fx, 0, cx},{0, fy, cy},{0, 0, 1}};
cv::Mat cameraMatrix = cv::Mat(3, 3, CV_32FC1, &cm);

double X,Y,Z;
for(int v = 0; v < HEIGHT; v++)
{
  for (int u = 0; u < WIDTH; u++)
  {
      if ( depth.at<uint16_t>(v,u) != 0 )
      {
            Z = depth.at<uint16_t>(v,u);

            cv::Mat src = cv::Mat(1,1,CV_32FC2), dst = cv::Mat(1,1,CV_32FC2);

            src.at<cv::Vec2f>(0,0)[0] = u;
            src.at<cv::Vec2f>(0,0)[1] = v;

            cv::undistortPoints(src,dst,cameraMatrix,distCoeffs);

            X = dst.at<cv::Vec2f>(0,0)[0] * Z;
            Y = dst.at<cv::Vec2f>(0,0)[1] * Z;

            // Point (X,Y,Z) in mm
      }
  }
}
```