Probabilistic Tracking using Stereo Cameras

Silvia-Laura Pintea *(6109969)*

<S.L.Pintea@student.uva.nl>

Contents

Chapter 1

Data Structure Documentation

1.1 annotationsHandle::ANNOTATION Struct Reference

A structure that stores a single annotation for a specific person.

Data Fields

- short int **id**
- cv::Point **location**
- vector< unsigned int > **poses**

1.2 annotationsHandle Class Reference

Class for annotating both positions and poses of the people in the images.

Data Structures

- struct ANNOTATION

  A structure that stores a single annotation for a specific person.

- struct ASSIGNED

  Shows which id from the old annotations is assigned to which id from the new annotations based on what minimal distance.

- struct FULL_ANNOTATIONS

  Structure containing a vector of annotations for each image.

Public Types

- enum POSE { SITTING, STANDING, BENDING, ORIENTATION }

  All considered poses.

Static Public Member Functions

- static void mouseHandlerAnn (int event, int x, int y, int flags, void *param)

  Mouse handler for annotating people's positions and poses.

- static void showMenu (cv::Point center)

  Draws the "menu" of possible poses for the current position.

- static int runAnn (int argc, char **argv)

  Starts the annotation of the images.

- static void trackbar_callback (int position, void *param)

  The "on change" handler for the track-bars.

- static void trackBarHandleFct (int position, void *param)

  A function that starts a new thread which handles the track-bar event.

- static void loadAnnotations (char *filename, vector< FULL_ANNOTATIONS > &loadedAnno)

  Load annotations from file.

- static void annoDifferences (vector< FULL_ANNOTATIONS > &train, vector< FULL_ANNOTATIONS > &test, double &avgDist, double &Ndiff, double avgOrientDiff, double poseDiff)

  Computes the average distance from the predicted location and the annotated one, the number of unpredicted people in each image and the differences in the pose estimation.

- static void correltateLocs (vector< ANNOTATION > &annoOld, vector< ANNOTATION > &annoNew, vector< ASSIGNED > &idAssignedTo)

  Correlate annotations' from locations in `annoOld` to locations in `annoNew` through IDs.

- static bool canBeAssigned (vector< ASSIGNED > &idAssignedTo, short int id, double newDist, short int to)

  Checks to see if a location can be assigned to a specific ID given the new distance.

- static void displayFullAnns (vector< FULL_ANNOTATIONS > &fullAnns)

  Displays the complete annotations for all images.

- static int runEvaluation (int argc, char **argv)

  Starts the annotation of the images.

1.2.1 Member Function Documentation

1.2.1.1 int runAnn (int *argc,* char ** *argv*) [static]

The parameters that need to be indicated are:

- argv[1] -- name of directory containing the images
- argv[2] -- the file contains the calibration data of the camera
- argv[3] -- the file in which the annotation data needs to be stored

1.2.1.2 int runEvaluation (int *argc,* char ** *argv*) [static]

The parameters that need to be indicated are:

- argv[1] -- train file with the correct annotations;
- argv[2] -- test file with predicted annotations;

1.3 annotationsHandle::ASSIGNED Struct Reference

Shows which id from the old annotations is assigned to which id from the new annotations based on what minimal distance.

Data Fields

- short int **id**
- short int **to**
- double **dist**

1.4 annotationsHandle::FULL_ANNOTATIONS Struct Reference

Structure containing a vector of annotations for each image.

Data Fields

- string **imgFile**
- vector< ANNOTATION > **annos**

Index