Eigen backgrounds – Documentation

University of Amsterdam

Contents

Chapter 1

Data Structure Documentation

1.1 AnnotatePos Class Reference

Class used for annotating human positions (coordinates) in the image.

Static Public Member Functions

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

  Creates priors for the hulls.

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

  Defines a mouse handler for defining the hull for each image.

- static int plotHull (IplImage *img, vector< CvPoint > &hull)

  Plots the hull corresponding to the coordinates stored in < <hull>>.

- static int plotAreaTmp (IplImage *src, float x, float y)

  Creates a copy of the image and displays the template corresponding to the position centered in <<x,y>>.

- static int plotArea (IplImage *img, float x, float y)

  Displays the template corresponding to the position centered in <<x,y>> on the original image.

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

  Depending on the mouse events i draws the templates.

- static int runAnn (int margc, char **margv)

  Starts the annotation process for the given parameters.

- static void loadAnnotations (const char *file, vector< positions_t > &locations)

  Reads locations from the given file and stores them in the variable `posT`.

- static float dist (const CvPoint &p1, const CvPoint &p2)

  Computes the Euclidian distance between two points.

- static void mergeAnnotations (const vector< vector< positions_t > > &loc, vector< positions_t > &res)

  Merges the locations of different people to a single position per each image.

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

  Runs the annotation merging process.

- static void averageDist (const vector< positions_t > &annotation, const vector< positions_t > &test, double &avgDist, double &Ndiff)

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

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

  Evaluates the annotations the and compare them by computing the average difference and the number of missed annotations.

Static Public Attributes

- static vnl_vector< FLOAT > logLocPrior

  Stores a location prior for the people's positions.

- static vector< CvPoint > prior

  Stores the "prior" points for the hull on the current image.

- static vector< CvPoint > locations

  Stores the coordinates of the center of the template for all annotations for each image.

- static parameter_t paramT

  Stores the image that is currently processed.

1.1.1 Member Function Documentation

1.1.1.1 void mouseHandlerAnn (int *event*, int *x*, int *y*, int *flags*, void * *param* = `NULL`) [static]

The mouse events can be:

- if the left button is pushed -- it draws a temporary template centered on the current pixel;
- if the mouse is moved and left button is down -- it plots and moves a temporary template;
- it the left button is up -- it stores the current location and it plots the final template.

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

The parameters have the following meanings:

- argv[1] -- the file containing the names of the images (relative paths);
- argv[2] -- the file containing the calibration data of the camera;
- argv[3] -- the file where the annotations need to be stored.

1.1.1.3 int runAnnMerging (int *argc*, char ** *argv*) [static]

The parameters that need to be given are:

- argv[1] -- camera height
- argv[2] -- person height
- argv[3] -- annotations directory
- argv[4] -- locations prior
- argv[5] -- merged annotations output
- argv[6+] -- annotations' files

1.1.1.4 int runAnnEvaluation (int *argc*, char ** *argv*) [static]

The input parameters are:

- argv[1] -- directory of annotations
- argv[2] -- calibration data
- argv[3] -- prior over locations
- argv[4] -- train annotations
- argv[5] -- test annotations

1.2 Background Class Reference

Public Member Functions

- Background (unsigned numImg, unsigned numVec)

  Constructor of the `Background` class -- initializes the variables of the class with the desired values.

- void processImage (const vnl_vector< FLOAT > &img)

  Adds the current image to the vector of images `imgs`; if the size is at maximum it removes the oldest image from the vector.

- void update ()

Computes the eigen-background of the current data stored in `imgs`; it considers only the best `noEigen`.

- void getBackground (const vnl_vector< FLOAT > &img, vnl_vector< FLOAT > &bg)

  If the image is one of the images out of which the eigen-space needs to be computed then it just resizes it, else it computes the back-projection of the image on the eigen-space and equalizes it.

- void segment (const vnl_vector< FLOAT > &img, vnl_vector< FLOAT > &bg, vnl_vector< FLOAT > &fg, std::vector< int > &mask)

  It computes the background model of an input image, it computes the foreground model and it also creates a mask having 1 on the positions of corresponding to the foreground pixels (the input image should not be out of the set used for generating the background model).

- void segment (const vnl_vector< FLOAT > &img, std::vector< int > &mask)

  It computes the background model for the input image, it computes a mask having 1 on the positions corresponding to the foreground pixels (the input image should not be out of the set used for generating the background model).

- void getProjection (const vnl_vector< FLOAT > &img, vnl_vector< FLOAT > &proj)

  Computes the projection of the image on the eigen-space (first the mean of the image needs to be subtracted).

- void xmlPack (XmlFile &f) const

  Writes the mean of the images used for the background model and the eigen-vectors into a xml file.

- void xmlUnpack (XmlFile &f)

  Reads the mean of the images used for the background model and the eigen- vectors from an xml file.

Data Fields

- unsigned N

  Represents the maximum number of images to be stored in the `imgs` vector.

- unsigned noEigen

  Represents the kept number of `eigen vectors` for an image.

- FLOAT n

  Represents the current number of images.

- std::list< vnl_vector< FLOAT > > imgs

  Represents the vector of stored images.

- std::vector< vnl_vector< FLOAT > > eigenvectors

  Represents the total eigen vector for an images.

- vnl_vector< FLOAT > sum

  A vector containing the sum of pixels for each image.

- vnl_vector< FLOAT > mu

  A vector containing the mean of pixels for each image.

Friends

- void getBackground (const char *filename, Background &bg)

  It either reads the background from a xml file or it generates it from the training data.

- void buildBackgroundModel (const char *outfile, Background &bg, char *trainList)

  A friend function of the class that builds a background model out of a training set and stores it in the indicated `outfile`.

1.3 findPersonImage Class Reference

Class used for tracking/finding people in the input images.

Public Member Functions

- **findPersonImage** (unsigned nVal)
- void stic ()

  Initializes `usec` *to the current value -- starts the timer.*

- unsigned long long stoc ()

  Shows the number of milliseconds passed since the timer was started.

- FLOAT logGaus (FLOAT sqDiff)

  Return the log-gaussian value for the input.

- void showImage (const char *win, const vnl_vector< FLOAT > &v)

  Converts the given image to the RGB color (CV_Luv2BGR) and displays it.

- void buildMasks ()

  Initializes the variable `masks` *with a corresponding mask centered at each pixel in the image.*

- FLOAT logMaskProbDiff (const vector< scanline_t > &(mask))

  For each pixel in the given mask computes the difference between the probability of that pixel being foreground and that pixel being background.

- FLOAT logMaskProb (FLOAT bgSum, const vector< scanline_t > &mask)

  Computes the difference between the sum of pixels probabilities for foreground and the sum of pixel probabilities for background.

- bool overlap (const vector< unsigned > &existing, int x, int y)

  Checks to see if two ground planes of a new mask and another existing mask overlap.

- void scanRest (vector< unsigned > &existing, vector< scanline_t > &existingMask, unsigned res, const vector< FLOAT > &logNumPrior, vector< vnl_vector< FLOAT > > &logPosProb, vector< FLOAT > &marginal, FLOAT &lSum)

  Recursively scan the image for a number of existing people that can be in the image; it chooses the location that increases the likelihood taking into account the existing positions.

- void scanArea (unsigned x1, unsigned y1, unsigned x2, unsigned y2, unsigned res, vnl_vector< FLOAT > &logPosProb, cv::Point &bestPoint, FLOAT &lSum)

  Searches for the positions in a given window [x1,x2,y1,y2] for the best position.

- cv::Point cvPoint (unsigned pos)

  Returns an Opencv point corresponding to the position with $x = (xmodwidth)$ *and* $y = (ydivwidth)$.

- FLOAT dist (const CvPoint &p1, const CvPoint &p2)

  Computes the euclidean distance between two points.

- void computeStats (vnl_vector< FLOAT > &img, const CvPoint &pos, FLOAT &sumR, FLOAT &sumR2, FLOAT &sumG, FLOAT &sumG2, FLOAT &sumB, FLOAT &sumB2, FLOAT &nPixels)

  For a given image and a position computes the sum of the pixel values in the corresponding mask for all 3 channels (R,G,B) and the squared sum of these values.

- void plotTrack (IplImage *img, const track_t &t, const unsigned idx, unsigned w)

  Function that plots the tracks from the tracking structure in the image.

- void doFindPerson (IplImage *src)

  Finds all possible locations containing people in a given image.

- int run (int argc, char **argv)

  Runs the person finder.

Data Fields

- vnl_vector< FLOAT > logLocPrior

  It is a vector storing the log-probabilities representing the location priors.

- vector< double > timestamps

  A vector containing the time-stamps of the images.

- vector< vector< unsigned long long > > times

  Stores the number of milliseconds needed to detect a a number of people.

- vnl_vector< FLOAT > logBGProb

  A vector containing the log-probabilities for the pixels being background.

- vnl_vector< FLOAT > logSumPixelBGProb

  The sum of log-probabilities for the pixels being background.

- vnl_vector< FLOAT > logFGProb

  Log-probabilities for the pixels in the image being foreground.

- vnl_vector< FLOAT > logSumPixelFGProb

  The sum of log-probabilities for the pixels being foreground.

- parameter_t paramT

A structure containing a pointer to an `IplImage`.

- vector< track_t > tracks

  A vector of structures storing the tracking information.

- unsigned N

  The maximum number of images to be stored in the vector used for building the background model.

- Background bgModel

  An instance of the class `Background` -- used to build the background/foreground model.

- unsigned long long usec

  Stores the current number of seconds and milliseconds.

- vector< vector< vector< scanline_t > > > masks

  Vector of masks (templates) for human positions.

- CvScalar trackCol [ ]

  Colors used for finding people.

1.3.1 Member Function Documentation

1.3.1.1 int run (int *argc*, char ** *argv*)

The arguments that need to be given are:

- argv[1] -- an xml file containing the background model
- argv[2] -- a file containing a list of image names
- argv[3] -- location priors
- argv[4] -- calibration data
- argv[5] -- the file name for the output information

1.4 findPersonStream Class Reference

Class used for tracking/finding people in the input images.

Public Member Functions

- **findPersonStream** (unsigned nVal)
- void stic ()

  Initializes `usec` to the current value -- starts the timer.

- unsigned long long stoc ()

  Shows the number of milliseconds passed since the timer was started.

- FLOAT logGaus (FLOAT sqDiff, FLOAT lpv2, FLOAT pv2)

  Return the log-gaussian value for the input.

- void buildMasks ()

  Initializes the variable `masks` with a corresponding mask centered at each pixel in the image.

- FLOAT logMaskProb (const vnl_vector< FLOAT > &logSumPixelBGProb, FLOAT bgSum, const vector< scanline_t > &mask)

  Computes the difference between the sum of pixels probabilities for foreground and the sum of pixel probabilities for background.

- bool overlap (const vector< unsigned > &existing, int x, int y)

  Checks to see if two ground planes of a new mask and another existing mask overlap.

- void scanRest (vector< unsigned > &existing, vector< scanline_t > &existingMask, unsigned res, const vnl_vector< FLOAT > &logSumPixelBGProb, vector< vnl_vector< FLOAT > > &logPosProb, vector< FLOAT > &marginal, FLOAT lSum)

  Recursively scan the image for a number of existing people that can be in the image; it chooses the location that increases the likelihood taking into account the existing positions.

- cv::Point cvPoint (unsigned pos)

  Returns an Opencv point corresponding to the position with $x = (xmodwidth)$ and $y = (ydivwidth)$.

- FLOAT dist (const CvPoint &p1, const CvPoint &p2)

  Computes the euclidean distance between two points.

- void plotTrack (IplImage *img, const track_t &t, const unsigned idx, unsigned w)

  Function that plots the tracks from the tracking structure in the image.

- void doFindPerson (unsigned imgNum, IplImage *src, const vnl_vector< FLOAT > &imgVec, vnl_vector< FLOAT > &bgVec, const FLOAT logBGProb, const vnl_vector< FLOAT > &logSumPixelBGProb)

  Finds all possible locations containing people in a given image.

- void plotHull (IplImage *img, vector< CvPoint > &hull)

  Plots the hull indicated by the parameter `hull` on the given image.

- void updateTracks (unsigned imgNum, const vector< unsigned > &positions)

  Given a set of new positions for a specific image, it assigns the best positions to the tracks.

- void initStaticProbs ()

  Initialize the prior probability over the number of people and the sum of pixels probabilities of the foreground pixels.

- void computeBGProb (const vnl_vector< FLOAT > &img, const vnl_vector< FLOAT > &bg, vnl_vector< FLOAT > &sumPixelProb, FLOAT &total)

  Computes the sum of background pixels' probabilities for the input image given a background model.

- void decode_stream ()

  Reads frames from an input streams, computes their background model and stores them in the buffer at the index indicated by the `currentIndex`.

- void synchronous_decode_stream ()

  Reads frames from an input streams, computes their background model and stores them in the buffer at the indexes 0 and 1.

- void showCopies ()

  Read the image at the index `currentIndex` from the buffer and "consumes" the image by storing it as the currently processed image and displaying its background model.

- void trackPeople ()

  Read the image from the buffer at the position indicated by `currentIndex` "consumes" the image by finding people's locations in the image and building the background model if needed.

- void synchronous_trackPeople ()

  It processed the images from the buffer as follows: it first reads the image on the position 0 in the buffer and processes it by finding people in it and building the background model if needed and then it does the same thing for the image on position 1 in the buffer.

- int run (int argc, char **argv)

  Tracks people in a video stream using a buffer in which it stores the frames that were not yet processed.

Data Fields

- Image_t * current

  An instance of the structure `Image_t` containing the currently processed image.

- Image_t buffers [2]

  A buffer containing 2 instances of the `Image_t`.

- int currentIndex

  The current index of the buffer in which the "decoded" images are stored.

- boost::mutex imgProtect

  A mutex to control the access to the buffer when images are written there.

- boost::mutex mutexA

  A `mutex` to control the writing to the buffer in the synchronous case (when images are written on positions 1 and 0 of the buffer).

- boost::mutex mutexB

  A `mutex` to control the writing to the buffer in the synchronous case (when images are written on positions 1 and 0 of the buffer).

- bool running

  Indicates if the `producer` of the class `ImgProducer` is running (reading frames, computing their background model and storing them in the buffer).

- vector< FLOAT > logNumPrior

  Represents a prior over the number of people in a frame.

- vector< CvPoint > priorHull

  A prior probability over the shape of the hull for the position of a person.

- int waitTime

  The time the process will wait for a key to be pressed.

- ImgProducer * producer

  A pointer to the class `ImgProducer`.

- vnl_vector< FLOAT > logLocPrior

  It is a vector storing the log-probabilities representing the location priors.

- vnl_vector< FLOAT > logSumPixelFGProb

  The sum of log-probabilities for the pixels being foreground.

- vector< track_t > tracks

  A vector of structures storing the tracking information.

- unsigned N

  The maximum number of images to be stored in the vector used for building the background model.

- Background bgModel

  An instance of the class `Background` -- used to build the background/foreground model.

- unsigned long long usec

  Stores the current number of seconds and milliseconds.

- vector< vector< vector< scanline_t > > > masks

  Vector of masks (templates) for human positions.

- CvScalar trackCol [ ]

  Colors used for different tracks.

1.4.1 Member Function Documentation

1.4.1.1 int run (int *argc*, char ** *argv*)

The input arguments are:

- argv[1] -- an stream file from which the frames are read
- argv[2] -- a background model or a training set for the background (?)
- argv[3] -- calibration data
- argv[4] -- location priors and hull priors

1.5 Image_t Struct Reference

Structure containing the IplImage processed, the background model, the sum of background pixels probabilities, the image probability.

Data Fields

- bool **consumed**
- unsigned **index**
- string **sourceName**
- IplImage * **img**
- vnl_vector< FLOAT > **imgVec**
- vnl_vector< FLOAT > **bgVec**
- vnl_vector< FLOAT > **logSumPixelBGProb**
- FLOAT **imgProb**

1.6 ImgProducer Class Reference

The `ImgProducer` class is used to read images from either a video stream or from a local directory.

Public Member Functions

- ImgProducer (const char *filename)

  The constructor of the class -- initializes the variables to the given values; if there is a video available, then it uses the capture from file (OpenCv), else it uses the files from the `filelist`.

- IplImage * getFrame ()

  If the `sourcetype` *is* `CAPTURE` *then it reads the next frame of the video, else if it is it loads the next image and returns it.*

- const std::string & getSource () const

  Returns either 'Not a file' or (in the case when the image names are read from the `filelist`*) the name of the next image to be processed.*

- void forward (unsigned frames)

  It updates the value of the variable `nextFrame` *-- with respect to the source type used (CAPTURE or FILELIST) -- such that the index of the next frame will indicate to the image position at* `frames` *forward from the current one.*

- void backward (unsigned frames)

  It updates the value of the variable `nextFrame` *-- with respect to the source type used (CAPTURE or FILELIST) -- such that the index of the next frame will indicate to the image position at* `frames` *backward from the current one.*

1.7 parameter_t Struct Reference

Stores the currently processed image as an IplImage.

Data Fields

- IplImage * **img**

1.8 positions_t Struct Reference

A structure containing the name of the image and a vector of locations.

Data Fields

- string **imgfile**
- vector< CvPoint > **loc**

1.9 scanline_t Struct Reference

Data Fields

- unsigned **line**
- unsigned **start**
- unsigned **end**

1.10 track_t Struct Reference

Structure used for tracking containing statistics about masks.

Data Fields

- vector< CvPoint > **positions**
- vector< unsigned > **imgID**
- FLOAT **sumR**
- FLOAT **sumG**
- FLOAT **sumB**
- FLOAT **sumR2**
- FLOAT **sumG2**
- FLOAT **sumB2**
- FLOAT **sumPix**

Index