Модуль выделения и прослеживания лиц на цифровых видеопоследовательностях

FD v. 1.0.0.0

Создано системой Doxygen 1.8.2

Чт 21 Мар 2013 17:43:09

Содержание

1 Общие сведения

1.1 Назначение

Динамически подключаемая библиотека FDaT (Face Detection and Tracking) предназначена для анализа и обработки цифровых видеопоследовательностей. Модуль обеспечивает регистрацию и прослеживание лиц на цифровом полутоновом изображении, получаемом от видеокамеры или из файла.

1.2 Ограничения применения библиотеки

• Линейное смещение лица за кадр не превышает половины соответствующего линейного размера.

1.3 Входные данные

• Видеопоток, размер кадра не менее 352x288 пикселей (монохром, 8 bit/px).

1.4 Выходные данные

• Список номеров лиц и координат их объемлющих прямоугольников;

1.5 Программный пакет

1. Динамически подключаемая библиотека iit-face-detector.dll для MS Windows XP, 2003 Server, Vista, 2008 Server, Windows 7, Windows 8.
2. Shared-object библиотека iit-face-detector.so для Linux Debian 5.0.
3. Заголовочные файлы iit-face-detector.h, VMDAParams.h.
4. Данное руководство разработчика.

1.6 История версий

1.6.1 Версия 1.0. (03/15/2013)

• Исходная версия библиотеки.

2 Иерархический список классов

2.1 Иерархия классов

Иерархия классов.

3 Список файлов

3.1 Файлы

Полный список файлов.

4 Классы

4.1 Структура CFACalibrationData

Данные калибровки камеры и разметки сцены, необходимые для измерения перспективы

#include <VmdaParams.h>

Открытые члены

- CFACalibrationData ()

  Конструктор по умолчанию для класса калибровки
- ~CFACalibrationData ()

  Деструктор по умолчанию для класса калибровки
- void CreateGrid (int ptsGridCount)

  Процедура класса для выделения памяти под опорные точки
- void CreateZ (int ptsZCount)

  Процедура класса для выделения памяти под вспомогательные точки
- void Destroy ()

  Процедура класса для освобождения памяти, выделенной процедурами CreateGrid(int ptsGridCount) и CreateZ(int ptsZCount)

Открытые атрибуты

- CFAPointXY * ptsGrid_3D

  Пространственные X-Y координаты опорных точек. Высоты (Z-координаты) всех точек считаются равными нулю
- CFAPointXY * ptsGrid_2D

  Пиксельные X-Y координаты опорных точек на изображении.
- CFAPointXY * ptsZ_2D

  Пиксельные X-Y координаты вспомогательных точек на изображении.
- int countPtsGrid

  Количество опорных точек
- int countPtsZ

  Количество вспомогательных точек, задающих эталонные высоты на кадре
- double postHeight

  Пространственная высота вспомогательных точек

4.1.1 Подробное описание

Данные калибровки камеры и разметки сцены, необходимые для измерения перспективы

4.1.2 Конструктор(ы)

4.1.2.1 CFACalibrationData::CFACalibrationData ( )

Конструктор по умолчанию для класса калибровки

4.1.2.2 CFACalibrationData::~CFACalibrationData ( )

Деструктор по умолчанию для класса калибровки

4.1.3 Методы

4.1.3.1 void CFACalibrationData::CreateGrid (int ptsGridCount)

Процедура класса для выделения памяти под опорные точки

Аргументы

ptsGridCount	- количество опорных точек 2D (координаты на изображении) и 3D (координаты в пространстве)

4.1.3.2 void CFACalibrationData::CreateZ (int ptsZCount)

Процедура класса для выделения памяти под вспомогательные точки

Аргументы

ptsZCount	- количество опорных точек 2D (координаты на изображении) данные должны быть сформированы так: нижняя точка эталона 1, верхняя точка эталона 1, нижняя точка эталона 2, верхняя точка эталона 2, ... нижняя точка эталона ptsZCount, верхняя точка эталона ptsZCount,

4.1.3.3 void CFACalibrationData::Destroy ( )

Процедура класса для освобождения памяти, выделенной процедурами CreateGrid(int ptsGridCount) и CreateZ(int ptsZCount)

Вызывается стандартным деструктором класса

4.1.4 Данные класса

4.1.4.1 int CFACalibrationData::countPtsGrid

Количество опорных точек

4.1.4.2 int CFACalibrationData::countPtsZ

Количество вспомогательных точек, задающих эталонные высоты на кадре

4.1.4.3 double CFACalibrationData::postHeight

Пространственная высота вспомогательных точек

4.1.4.4 CFAPointXY* CFACalibrationData::ptsGrid_2D

Пиксельные X-Y координаты опорных точек на изображении.

Измеряются в долях от размера изображения [0..1]

4.1.4.5 CFAPointXY* CFACalibrationData::ptsGrid_3D

Пространственные X-Y координаты опорных точек. Высоты (Z-координаты) всех точек считаются равными нулю

4.1.4.6 CFAPointXY* CFACalibrationData::ptsZ_2D

Пиксельные X-Y координаты вспомогательных точек на изображении.

Измеряются в долях от размера изображения [0..1]

Объявления и описания членов структуры находятся в файле:

- VmdaParams.h

4.2 Структура CFAPointXY

Точка с двумерными вещественными координатами

#include <FDaTParams.h>

Открытые члены

- CFAPointXY ()
  конструктор по умолчанию для точки на кадре
- CFAPointXY (double _x, double _y)
  альтернативный конструктор для точки на кадре

Открытые атрибуты

- double x
  X-координата точки
- double y
  Y-координата точки

4.2.1 Подробное описание

Точка с двумерными вещественными координатами

4.2.2 Конструктор(ы)

4.2.2.1 CFAPointXY::CFAPointXY ( )

конструктор по умолчанию для точки на кадре

4.2.2.2 CFAPointXY::CFAPointXY (double _x, double _y)

альтернативный конструктор для точки на кадре

Аргументы

_x	- вещественная координата X
_y	- вещественная координата Y

4.2.3 Данные класса

4.2.3.1 double CFAPointXY::x

X-координата точки

в двумерном случае - на кадре, тогда измеряется в процентах от ширины кадра [0.0..1.0] в трехмерном случае - в пространстве (высота считается равной нулю или известной)

4.2.3.2 double CFAPointXY::y

Y-координата точки

в двумерном случае - на кадре, тогда измеряется в процентах от ширины кадра [0.0..1.0] в трехмерном случае - в пространстве (высота считается равной нулю или известной)

Объявления и описания членов структуры находятся в файле:

- FDaTParams.h

4.3 Структура CFAPointXYZ

Точка с трехмерными вещественными координатами

#include <FDaTParams.h>

Открытые члены

- CFAPointXYZ ()
  конструктор по умолчанию для точки на кадре
- CFAPointXYZ (double _x, double _y, double _z)
  альтернативный конструктор для точки на кадре

Открытые атрибуты

- double x
  X-координата точки в пространстве
- double y
  Y-координата точки в пространстве
- double z
  Z-координата точки в пространстве

4.3.1 Подробное описание

Точка с трехмерными вещественными координатами

4.3.2 Конструктор(ы)

4.3.2.1 CFAPointXYZ::CFAPointXYZ ( )

конструктор по умолчанию для точки на кадре

4.3.2.2 CFAPointXYZ::CFAPointXYZ (double _x, double _y, double _z)

альтернативный конструктор для точки на кадре

Аргументы

_x	- вещественная координата X
_y	- вещественная координата Y
_z	- вещественная координата Z

4.3.3 Данные класса

4.3.3.1 double CFAPointXYZ::x

X-координата точки в пространстве

4.3.3.2 double CFAPointXYZ::y

Y-координата точки в пространстве

4.3.3.3 double CFAPointXYZ::z

Z-координата точки в пространстве

Объявления и описания членов структуры находятся в файле:

- FDaTParams.h

4.4 Структура CFAPolyline

Ломаная линия с заданным числом точек излома, используемая для ограничения некоторого многогранника.

#include <FDaTParams.h>

Открытые члены

- CFAPolyline ()

  Конструктор по умолчанию
- ~CFAPolyline ()

  Деструктор по умолчанию
- void Create (int N)

  Процедура класса для выделения памяти для точек ломаной
- void Destroy ()

  Процедура класса для освобождения памяти, выделенной под точки ломаной процедурой Create(int N)

Открытые атрибуты

- CFAPointXY * points

  Точки ломаной, измеряемые в процентах от размеров кадра [0.0..1.0].
- unsigned int count

  Количество точек (вершин многогранника)

4.4.1 Подробное описание

Ломаная линия с заданным числом точек излома, используемая для ограничения некоторого многогранника.

Последняя точка автоматически соединяется с первой.

4.4.2 Конструктор(ы)

4.4.2.1 CFAPolyline::CFAPolyline ( )

Конструктор по умолчанию

4.4.2.2 CFAPolyline::~CFAPolyline ( )

Деструктор по умолчанию

4.4.3 Методы

4.4.3.1 void CFAPolyline::Create (int N)

Процедура класса для выделения памяти для точек ломаной

Аргументы

N	- количество точек ломаной

4.4.3.2 void CFAPolyline::Destroy ( )

Процедура класса для освобождения памяти, выделенной под точки ломаной процедурой Create(int N)

Вызывается стандартным деструктором класса

4.4.4 Данные класса

4.4.4.1 unsigned int CFAPolyline::count

Количество точек (вершин многогранника)

4.4.4.2 CFAPointXY* CFAPolyline::points

Точки ломаной, измеряемые в процентах от размеров кадра [0.0..1.0].

Объявления и описания членов структуры находятся в файле:

- FDaTParams.h

4.5 Структура CFARegionOfInterest

Области интереса и области игнорирования кадра.

#include <FDaTParams.h>

Открытые члены

- CFARegionOfInterest ()
- ~CFARegionOfInterest ()

  Деструктор класса по умолчанию
- void CreateInclude (int N)

  Процедура класса для выделения памяти под области интереса
- void DeleteInclude ()

  Процедура класса для освобождения памяти, выделенной процедурой CreateInclude(int N)
- void CreateExclude (int N)

Процедура класса для выделения памяти под области игнорирования

- void DeleteExclude ()

  Процедура класса для освобождения памяти, выделенной процедурой CreateExclude(int N)

Открытые атрибуты

- unsigned int includeCount

  Число областей интереса
- CFAPolyline * includeRegions

  Области интереса, координаты которых измеряются в процентах от размеров кадра
- unsigned int excludeCount

  Число областей игнорирования
- CFAPolyline * excludeRegions

  Области игнорирования, координаты которых измеряются в процентах от размеров кадра

4.5.1 Подробное описание

Области интереса и области игнорирования кадра.

Если область игнорирования пересекается с областью интереса, то предпочтение отдается области игнорирования.

4.5.2 Конструктор(ы)

4.5.2.1 CFARegionOfInterest::CFARegionOfInterest ( )

4.5.2.2 CFARegionOfInterest::~CFARegionOfInterest ( )

Деструктор класса по умолчанию

4.5.3 Методы

4.5.3.1 void CFARegionOfInterest::CreateExclude (int N)

Процедура класса для выделения памяти под области игнорирования

Аргументы

N	- количество областей игнорирования

4.5.3.2 void CFARegionOfInterest::CreateInclude (int N)

Процедура класса для выделения памяти под области интереса

Аргументы

N	- количество областей интереса

4.5.3.3 void CFARegionOfInterest::DeleteExclude ( )

Процедура класса для освобождения памяти, выделенной процедурой CreateExclude(int N)

НЕ вызывается стандартным деструктором класса, однако вызывается при вызове метода Destroy() класса SFDParams1. Также удаляет память для всех точек, входящих в области игнорирования

4.5.3.4 void CFARegionOfInterest::DeleteInclude ( )

Процедура класса для освобождения памяти, выделенной процедурой CreateInclude(int N)

НЕ вызывается стандартным деструктором класса, нужно вызывать отдельно. Также удаляет память для всех точек, входящих в области интереса

4.5.4 Данные класса

4.5.4.1 unsigned int CFARegionOfInterest::excludeCount

Число областей игнорирования

4.5.4.2 CFAPolyline* CFARegionOfInterest::excludeRegions

Области игнорирования, координаты которых измеряются в процентах от размеров кадра

4.5.4.3 unsigned int CFARegionOfInterest::includeCount

Число областей интереса

4.5.4.4 CFAPolyline* CFARegionOfInterest::includeRegions

Области интереса, координаты которых измеряются в процентах от размеров кадра

Объявления и описания членов структуры находятся в файле:

- FDaTParams.h

4.6 Структура CFASizeConstraint

Ограничения на размер детектируемого лица

#include <FDaTParams.h>

Открытые члены

- CFASizeConstraint ()
  Конструктор по умолчанию для класса ограничений

Открытые атрибуты

- double minSize
  Минимальный размер лица
- double maxSize
  Максимальный размер лица

4.6.1 Подробное описание

Ограничения на размер детектируемого лица

4.6.2 Конструктор(ы)

4.6.2.1 CFASizeConstraint::CFASizeConstraint ( )

Конструктор по умолчанию для класса ограничений

4.6.3 Данные класса

4.6.3.1 double CFASizeConstraint::maxSize

Максимальный размер лица

Измеряется в процентах от площади кадра

4.6.3.2 double CFASizeConstraint::minSize

Минимальный размер лица

Измеряется в процентах от площади кадра

Объявления и описания членов структуры находятся в файле:

- FDaTParams.h

4.7 Структура FaceRect

#include <iit-face-detector.h>

Открытые атрибуты

- int X
- int Y
- int W
- int H

4.7.1 Данные класса

4.7.1.1 int FaceRect::H

4.7.1.2 int FaceRect::W

4.7.1.3 int FaceRect::X

4.7.1.4 int FaceRect::Y

Объявления и описания членов структуры находятся в файле:

- iit-face-detector.h

4.8 Структура FD_IRecognizer

Интерфейс на стороне библиотеки обнаружения лиц, вызываемый клиентским приложением

#include <iit-face-detector.h>

Открытые члены

- virtual void OnFrame8bit (unsigned int nStride, double dtPosixTime, const unsigned char *frameData)=0

  Передача нового кадра анализатору. Каждый пиксел кадра задаётся 8 битами
- virtual void OnFrame24bit_YUV (unsigned int nStride, double dtPosixTime, const unsigned char *frameDataY, const unsigned char *frameDataU, const unsigned char *frameDataV)=0

  Передача нового кадра. Каждый пиксел кадра задаётся 24 битами в формате YUV (в версии 1.0 работает только компонента Y)

- virtual void Flush ()=0

  Команда анализатору на завершение записи всех лиц.
- virtual void Renew ()=0

  Очистка внутренней информации библиотеки обо всех имеющихся лицах.
- virtual void Destroy ()=0

  Удаление объекта, реализующего интерфейс детектора
- virtual void SwapParams (const SFDaTParams *pParams)=0

  Замена параметров детектирования, не останавливая процесс прослеживания

4.8.1 Подробное описание

Интерфейс на стороне библиотеки обнаружения лиц, вызываемый клиентским приложением

Данный интерфейс реализуется библиотекой-анализатором, его методы вызываются клиентским приложением и используются для передачи данных и управления анализатором

4.8.2 Методы

4.8.2.1 virtual void FD_IRecognizer::Destroy ( ) [pure virtual]

Удаление объекта, реализующего интерфейс детектора

4.8.2.2 virtual void FD_IRecognizer::Flush ( ) [pure virtual]

Команда анализатору на завершение записи всех лиц.

Для каждого лица, которое было создано с помощью CreateObject анализатор вызывает либо CommitObject, либо UndoObject

4.8.2.3 virtual void FD_IRecognizer::OnFrame24bit_YUV (unsigned int nStride, double dtPosixTime, const unsigned char * frameDataY, const unsigned char * frameDataU, const unsigned char * frameDataV) [pure virtual]

Передача нового кадра. Каждый пиксел кадра задаётся 24 битами в формате YUV (в версии 1.0 работает только компонента Y)

Аргументы

nStride	величина страйда поступившего кадра
dtPosixTime	момент времени поступления кадра,
frameDataY	указатель на массив байт Y-компоненты текущего кадра
frameDataU	указатель на массив байт U-компоненты текущего кадра
frameDataV	указатель на массив байт V-компоненты текущего кадра

4.8.2.4 virtual void FD_IRecognizer::OnFrame8bit (unsigned int nStride, double dtPosixTime, const unsigned char * frameData) [pure virtual]

Передача нового кадра анализатору. Каждый пиксел кадра задаётся 8 битами

Аргументы

nStride	величина страйда поступившего кадра
dtPosixTime	момент времени поступления кадра,
frameData	указатель на массив байт яркостей текущего кадра

4.8.2.5 virtual void FD_IRecognizer::Renew ( ) [pure virtual]

Очистка внутренней информации библиотеки обо всех имеющихся лицах.

Если в данный момент анализатор отслеживал какие-либо лица, то для них необходимо получить новые ID у коллектора

4.8.2.6 virtual void FD_IRecognizer::SwapParams (const SFDaTParams * pParams) [pure virtual]

Замена параметров детектирования, не останавливая процесс прослеживания

Меняет все параметры детектора лиц "на лету"

Аргументы

pParams	указатель на структуру с новыми параметрами детектора

Объявления и описания членов структуры находятся в файле:

- iit-face-detector.h

4.9 Структура FD_IResultsCollector

Интерфейс на стороне клиентского приложения, вызываемый библиотекой обнаружения лиц

#include <iit-face-detector.h>

Открытые члены

- virtual int CreateObject ()=0
- virtual void SetGeometry (int objectId, double x, double y, double width, double height)=0

  Установка пиксельных координат центра масс и размеров лица с данным ID на текущем кадре.
- virtual void CommitObject (int objectId)=0

  Удаление лица с данным ID с записью трека в базу.
- virtual void UndoObject (int objectId)=0

  Удаление лица с данным ID с удалением трека из базы.
- virtual bool IsFull () const =0

  Проверка заполненности коллектора объектов.

4.9.1 Подробное описание

Интерфейс на стороне клиентского приложения, вызываемый библиотекой обнаружения лиц

Данный интерфейс реализуется клиентским приложением и предоставляется библиотеке-анализатору для возвращения результатов детектирования лиц

4.9.2 Методы

4.9.2.1 virtual void FD_IResultsCollector::CommitObject (int objectId) [pure virtual]

Удаление лица с данным ID с записью трека в базу.

Т.е. данная процедура сообщает приложению, что лицо с данным ID больше отслеживаться не будет.

Аргументы

objectId	ID удаляемого лица

4.9.2.2 virtual int FD_IResultsCollector::CreateObject ( ) [pure virtual]

Создание нового лица и начало записи его трека.

Возвращает

ID номер нового лица

4.9.2.3 virtual bool FD_IResultsCollector::IsFull ( ) const [pure virtual]

Проверка заполненности коллектора объектов.

Возвращает

если результат равен true, то коллектор больше не принимает данные о лицах. SetGeometry и SetGeometry3D в данном случае не вызываются до момента, пока коллектор не освободится

4.9.2.4 virtual void FD_IResultsCollector::SetGeometry (int objectId, double x, double y, double width, double height) [pure virtual]

Установка пиксельных координат центра масс и размеров лица с данным ID на текущем кадре.

Передача координат лица с ID объекта, равным objectId. Если эта функция не вызывалась несколько кадров и лицо не было удалено из базы, считается, что лицо не изменяло своих координат на этих кадрах.

Аргументы

objectId	ID лица, для которого устанавливаются координаты и размеры
x	пиксельная X-координата центра лица на изображении (в долях от ширины изображения [0..1])
y	пиксельная Y-координата центра лица на изображении (в долях от высоты изображения [0..1])
width	пиксельная ширина лица на изображении (в долях от ширины изображения [0..1])
height	пиксельная высота лица на изображении (в долях от высоты изображения [0..1])

4.9.2.5 virtual void FD_IResultsCollector::UndoObject (int objectId) [pure virtual]

Удаление лица с данным ID с удалением трека из базы.

Т.е. данная процедура сообщает приложению, что лицо с данным ID на самом деле не являлось лицом.

Аргументы

objectId	ID удаляемого лица

Объявления и описания членов структуры находятся в файле:

- iit-face-detector.h

4.10 Структура SFDaTParams

Базовый класс параметров детектора

#include <iit-face-detector.h>

Производные классы:SFDaTParams1.

Открытые члены

- SFDaTParams ()
  Конструктор по умолчанию

Открытые атрибуты

- unsigned int nSize
  Размер текущего класса параметров
- unsigned int nVersion
  Версия текущего класса параметров

4.10.1 Подробное описание

Базовый класс параметров детектора

4.10.2 Конструктор(ы)

4.10.2.1 SFDaTParams::SFDaTParams ( )

Конструктор по умолчанию

4.10.3 Данные класса

4.10.3.1 unsigned int SFDaTParams::nSize

Размер текущего класса параметров

4.10.3.2 unsigned int SFDaTParams::nVersion

Версия текущего класса параметров

Объявления и описания членов структуры находятся в файле:

- iit-face-detector.h

4.11 Структура SFDaTParams1

Основной класс параметров текущей версии детектора лиц

#include <FDaTParams.h>

Базовые классы:SFDaTParams.

Открытые члены

Основные процедуры

- SFDaTParams1 ()
  Конструктор по умолчанию, определяющий дефолтные значения всех параметров
- void Destroy ()
  Процедура удаления памяти, выделенной под параметры

Открытые атрибуты

Общие параметры

- CFARegionOfInterest roi
  Области интереса и игнорирования

Параметры алгоритма трекирования лиц

- CFASizeConstraint trackerWidthConstraint
  Ограничения на ширину лица
- CFASizeConstraint trackerHeightConstraint
  Ограничения на высоту лица
- double timeUntilLost
  Время в секундах до удаления пропавшего лица из базы

Режимы работы

- int useAntiShaker
  Использовать ли алгоритм устранения "дрожания" камеры. Активны режим 0 - неподвижная камера и 1 - дрожащая камера
- bool skipEqualFrames
  Пропускать (true) или нет (false) одинаковые кадры.

4.11.1 Подробное описание

Основной класс параметров текущей версии детектора лиц

4.11.2 Конструктор(ы)

4.11.2.1 SFDaTParams1::SFDaTParams1 ( )

Конструктор по умолчанию, определяющий дефолтные значения всех параметров

4.11.3 Методы

4.11.3.1 void SFDaTParams1::Destroy ( )

Процедура удаления памяти, выделенной под параметры

4.11.4 Данные класса

4.11.4.1 CFARegionOfInterest SFDaTParams1::roi

Области интереса и игнорирования

4.11.4.2 bool SFDaTParams1::skipEqualFrames

Пропускать (true) или нет (false) одинаковые кадры.

4.11.4.3 double SFDaTParams1::timeUntilLost

Время в секундах до удаления пропавшего лица из базы

4.11.4.4 CFASizeConstraint SFDaTParams1::trackerHeightConstraint

Ограничения на высоту лица

4.11.4.5 CFASizeConstraint SFDaTParams1::trackerWidthConstraint

Ограничения на ширину лица

4.11.4.6 int SFDaTParams1::useAntiShaker

Использовать ли алгоритм устранения "дрожания" камеры. Активны режим 0 - неподвижная камера и 1 - дрожащая камера

Объявления и описания членов структуры находятся в файле:

- FDaTParams.h

5 Файлы

5.1 Файл FDaTParams.h

#include "iit-face-detector.h"

Классы

- struct CFAPointXY

  Точка с двумерными вещественными координатами
- struct CFAPointXYZ

  Точка с трехмерными вещественными координатами
- struct CFAPolyline

  Ломаная линия с заданным числом точек излома, используемая для ограничения некоторого многогранника.
- struct CFASizeConstraint

  Ограничения на размер детектируемого лица
- struct CFARegionOfInterest

  Области интереса и области игнорирования кадра.
- struct SFDaTParams1

  Основной класс параметров текущей версии детектора лиц

Макросы

- #define _VMDA_PARAMS4_IIT_H_

5.1.1 Макросы

5.1.1.1 #define _VMDA_PARAMS4_IIT_H_

5.2 Файл iit-face-detector.h

#include <vector>

Классы

- struct SFDaTParams

  Базовый класс параметров детектора

- struct FaceRect
- struct FD_IResultsCollector

  Интерфейс на стороне клиентского приложения, вызываемый библиотекой обнаружения лиц
- struct FD_IRecognizer

  Интерфейс на стороне библиотеки обнаружения лиц, вызываемый клиентским приложением

Макросы

- #define IITEXPORT extern "C"

Функции

- asm (".section .drectve")

  функция для экспорта библиотеки в виде shared object в ОС Linux.
- asm (".ascii \"-export:FD_CreateRecognizer\"")

  функция создания библиотеки в ОС Linux.
- IITEXPORT FD_IRecognizer * FD_CreateRecognizer (int nFrameWidth, int nFrameHeight, FD_IResultsCollector *pMDCollector, const SFDaTParams *pParams, char *szErrDescBuf, int nErrDescBufLength)
- IITEXPORT void DetectFaces8bit (unsigned int nWidth, unsigned int nHeight, const unsigned char *pData, unsigned int nStride, unsigned int nMinfaceSize, unsigned int nMaxFaceSize, bool fast, FaceRect *pfaceRects, int &nCount)

  функция, выделяющая все лица на отдельном 8-битном кадре

5.2.1 Макросы

5.2.1.1 #define IITEXPORT extern "C"

5.2.2 Функции

5.2.2.1 asm (".section .drectve")

функция для экспорта библиотеки в виде shared object в ОС Linux.

5.2.2.2 asm (".ascii \"-export:FD_CreateRecognizer\"")

функция создания библиотеки в ОС Linux.

5.2.2.3 IITEXPORT void DetectFaces8bit (unsigned int nWidth, unsigned int nHeight, const unsigned char * pData, unsigned int nStride, unsigned int nMinfaceSize, unsigned int nMaxFaceSize, bool fast, FaceRect * pfaceRects, int & nCount)

функция, выделяющая все лица на отдельном 8-битном кадре

Аргументы

nWidth	ширина изображения в пикселах
nHeight	высота изображения в пикселах
pData	указатель на данные
nStride	межстрочный шаг
nMinfaceSize	минимальный размер детектируемых лиц в пикселах
nMaxFaceSize	максимальный размер детектируемых лиц в пикселах
fast	включение быстрого режима
pfaceRects	буфер с найденными лицами
nCount	количество найденных лиц

5.2.2.4 IITEXPORT FD_IRecognizer* FD_CreateRecognizer (int nFrameWidth, int nFrameHeight, FD_IResultsCollector * pMDCollector, const SFDaTParams * pParams, char * szErrDescBuf, int nErrDescBufLength)

функция, создающая библиотеку с параметрами pParams

Возвращает

указатель на экземпляр класса FD_IRecognizer

Аргументы

nFrameWidth	ширина кадра (в пикселях)
nFrameHeight	высота кадра (в пикселях)
pMDCollector	указатель на коллектор движущихся объектов
pParams	указатель на параметры детектора лиц. Допустимо передать 0 для использования параметров по умолчанию
szErrDescBuf	указатель на буфер описания ошибки
nErrDescBuf-Length	длина буфера описания ошибки

6 Примеры работы с библиотекой

6.1 Пример работы с объектом

Примерный сценарий добавления объекта в коллектор:

```
{пришел кадр №1, обнаружилось лицо в точке (0.1; 0.1)}
int objectId = pCollector->CreateObject(); \n
pCollector->SetGeometry(objectId, 0.1, 0.1, 0.2, 0.2);


{пришел кадр №2, лицо переместилось в точку (0.2; 0.2)}
pCollector->SetGeometry(objectId, 0.2, 0.2, 0.2, 0.2);


{пришел кадр №3, лицо переместилось в точку (0.4; 0.2)}
pCollector->SetGeometry(objectId, 0.4, 0.2, 0.2, 0.2);


{пришел кадр №4, лицо исчезло}
if(объект на самом деле не являлся лицом)
        pCollector->UndoObject(objectId);
else
        pCollector->CommitObject(objectId);
```

6.2 Рекомендации по настройкам детекторов

• Для инициализации библиотеки используются функции CreateRecognizer. В зультате вызова создаются внутренние объекты библиотеки. Функция CreateRecognizer должна быть вызвана прежде любой другой функции библиотеки.

• Минимальные линейные размеры лица, который детектируется алгоритмом, равны 24 пикселям.

• Максимальная скорость детектируемого лица не более половины линейных размеров в пикселях/кадр.

• Скорость работы библиотеки очень сильно (и нелинейно) зависит от настроенных минимальных размеров лица.

• Области интереса используются для выделения областей слежения от областей с шумами (деревья, отражения, волны). Области игнорирования используются для выделения шумов из больших областей интереса.

Предметный указатель