Модуль выделения и прослеживания движущихся объектов на цифровых видеопоследовательностях

VMDA v. 9.2.0.0

Содержание

1 Общие сведения

1.1 Назначение

Динамически подключаемая библиотека VMDA (Video Motion Detection and Analysis) предназначена для анализа и обработки цифровых видеопоследовательностей. Модуль обеспечивает регистра-

цию факта движения, выделение и прослеживание движущихся объектов на цифровом полутоновом изображении, а также обнаружение появившихся или оставленных предметов получаемом от видеокамеры или из файла.

1.2 Ограничения применения библиотеки

• Скорость объектов предполагается локально постоянной и больше 1 пикс/кадр.

• Zoom камеры отключен.

• Камера стационарна.

1.3 Входные данные

• Видеопоток, размер кадра не менее 352x288 пикселей (монохром, 8 bit/px или YUV, 24 bit/px);

1.4 Выходные данные

• Список номеров объектов и координат их объемлющих прямоугольников;

• Для каждого объекта его тип (человек, машина, все остальное; предмет появившийся, исчезнувший);

1.5 Программный пакет

1. Динамически подключаемая библиотека iit-detector.dll для MS Windows XP, 2003 Server, Vista, 2008 Server, Windows 7.
2. Shared-object библиотека simple_detector.so для Linux Debian 5.0.
3. Заголовочные файлы iit-detector.h, VMDAParams4.h.
4. Данное руководство разработчика.

1.6 История версий

1.6.1 Версия 9.2 (05/03/2012)

• Добавлены интеллектуальные статистические яркостно-цветовые корреляторы для восстановления треков объектов после потери или коллизии.

• Добавлена работа с разметкой сцены для преобразования 2D-координат кадра в 3D-координаты сцены.

• Улучшена производительность всех модулей в целом.

1.6.2 Версия 9.0 (03/15/2012)

• Добавлена возможность работы с цветом.

• Добавлено определение типа движущегося объекта (человек/машина/все остальное).

• Добавлено определение типа предмета (появившийся/исчезнувший).

• Созданы алгоритмы отсечения ложных срабатываний детектора.

• SVmdaParams3 структура улучшена до SVmdaParams4.

• Добавлена возможность подсчета количества людей в группе (работает совместно с детектором)

1.6.3 Версия 8.0 (07/07/2010)

• Значительно улучшен алгоритм NOMOD (New Or Missing Object Detector).

• Значительно улучшены качество и быстродействие обработки антишейкера.

• Новые разделённые интерфейсы NOMOD и MD (необходимо обновление интерфейса обратной связи).

1.6.4 Версия 7.6. (03/10/2010)

• Новый алгоритм автоматической чувствительности.

• Улучшен алгоритм слежения за объектами.

• Улучшена фильтрация траектории объектов.

• SVmdaParams2 структура улучшена до SVmdaParams3.

• Добавлена функция SwapParams для применения новых параметров в ходе выполнения программы «на лету».

1.6.5 Версия 7.0. (07/15/2009)

• Новый многозонный алгоритм слежения.

• Добавлена поддержка NVIDIA CUDA (требуется дополнительная библиотека и наличие ПК ускорителя NVIDIA GeForce 7600GT и новее) для ускорения HD-слежения с использованием графического процессора NVIDIA.

• Выпущена версия для процессоров ARM.

2 Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

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 <VMDAParams4.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:

- VMDAParams4.h

4.2 Структура CFAPointXY

Точка с двумерными вещественными координатами

#include <VMDAParams4.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:

- VMDAParams4.h

4.3 Структура CFAPointXYZ

Точка с трехмерными вещественными координатами

#include <VMDAParams4.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:

- VMDAParams4.h

4.4 Структура CFAPolyline

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

#include <VMDAParams4.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:

- VMDAParams4.h

4.5 Структура CFARegionOfInterest

Области интереса и области игнорирования кадра.

#include <VMDAParams4.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 SVMDAParams4. 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:

- VMDAParams4.h

4.6 Структура CFASizeConstraint

Ограничения на размер детектируемого объекта

#include <VMDAParams4.h>

Открытые члены

- CFASizeConstraint ()

  Конструктор по умолчанию для класса ограничений

Открытые атрибуты

- double minSize

Минимальный размер объекта

- double maxSize

  Максимальный размер объекта

4.6.1 Подробное описание

Ограничения на размер детектируемого объекта

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 объекта

Измеряется в процентах от площади кадра или в пространстве при включенных 3D-измерениях

4.6.3.2 double CFASizeConstraint::minSize

Минимальный размер объекта

Измеряется в процентах от площади кадра или в пространстве при включенных 3D-измерениях

Объявления и описания членов структуры находятся в файле:

- VMDAParams4.h

4.7 Структура SVmdaParams

Базовый класс параметров детектора

#include <iit-detector.h>

Производные классы:SVmdaParams4.

Открытые члены

- SVmdaParams ()

  Конструктор по умолчанию

Открытые атрибуты

- unsigned int nSize

  Размер текущего класса параметров
- unsigned int nVersion

  Версия текущего класса параметров

4.7.1 Подробное описание

Базовый класс параметров детектора

4.7.2 Конструктор(ы)

4.7.2.1 SVmdaParams::SVmdaParams ( )

Конструктор по умолчанию

4.7.3 Данные класса

4.7.3.1 unsigned int SVmdaParams::nSize

Размер текущего класса параметров

4.7.3.2 unsigned int SVmdaParams::nVersion

Версия текущего класса параметров

Объявления и описания членов структуры находятся в файле:

- iit-detector.h

4.8 Структура SVmdaParams4

Основной класс параметров текущей версии детектора

#include <VMDAParams4.h>

Базовые классы:SVmdaParams.

Открытые члены

Основные процедуры

- SVmdaParams4 ()
  Конструктор по умолчанию, определяющий дефолтные значения всех параметров
- void Destroy ()
  Процедура удаления памяти, выделенной под параметры

Открытые атрибуты

Общие параметры

- CFACalibrationData calibrationData
  Данные внешней калибровки камеры для текущей сцены
- CFARegionOfInterest roi
  Интересующая нас (в плане трекинга и оставленных предметов) область поля зрения

Параметры алгоритма трекирования целей

- CFASizeConstraint trackerWidthConstraint
  Ограничения на ширину движущегося объекта
- CFASizeConstraint trackerHeightConstraint
  Ограничения на высоту движущегося объекта
- double trackerSensitivity
  Чувствительность детектора движения [0...100].
- double timeUntilLost
  Время в секундах до удаления пропавшего объекта из базы

Параметры алгоритма обнаружения появившихся/исчезнувших предметов

- CFASizeConstraint unattendedObjectWidthConstraint
  Ограничения на ширину появившегося/исчезнувшего предмета
- CFASizeConstraint unattendedObjectHeightConstraint
  Ограничения на высоту появившегося/исчезнувшего предмета
- double unattendedObjectSensitivity
  Чувствительность детектора появившихся/исчезнувших предметов [1...25].

Режимы работы

- bool detectUnattendedObjects
  Детектировать (true) или нет (false) появившиеся или исчезнувшие предметы.
- 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.
- unsigned int filterStrength
  Число кадров [10...50] для фильтрации движения.
- bool useCalibrationData
  Использовать ли перспективные измерения
- bool singleRegion
  Использовать (true) или нет (false) 3D измерения по размеченной сцене

Дополнительная обработка объектов

- bool determineGivenTaken
  Использовать ли определение на принесённый\унесённый объект
- bool determineHumanCar
  Использовать ли алгоритм распознавания человека и машины
- bool determineHumanCount
  Использовать ли алгоритмы определения количества людей в группе
- bool determineNoise
  Использовать ли алгоритмы отсечения псевдодвижущихся объектов
- bool determineColorIdentity
  Использовать ли алгоритмы обнаружения соответствия объектов по их цветовым характеристикам

4.8.1 Подробное описание

Основной класс параметров текущей версии детектора

4.8.2 Конструктор(ы)

4.8.2.1 SVmdaParams4::SVmdaParams4 ( )

Конструктор по умолчанию, определяющий дефолтные значения всех параметров

4.8.3 Методы

4.8.3.1 void SVmdaParams4::Destroy ( )

Процедура удаления памяти, выделенной под параметры

4.8.4 Данные класса

4.8.4.1 CFACalibrationData SVmdaParams4::calibrationData

Данные внешней калибровки камеры для текущей сцены

4.8.4.2 bool SVmdaParams4::detectUnattendedObjects

Детектировать (true) или нет (false) появившиеся или исчезнувшие предметы.

4.8.4.3 bool SVmdaParams4::determineColorIdentity

Использовать ли алгоритмы обнаружения соответствия объектов по их цветовым характеристикам

4.8.4.4 bool SVmdaParams4::determineGivenTaken

Использовать ли определение на принесённый\унесённый объект

Работает только при включённом обнаружении оставленных предметов.

4.8.4.5 bool SVmdaParams4::determineHumanCar

Использовать ли алгоритм распознавания человека и машины

4.8.4.6 bool SVmdaParams4::determineHumanCount

Использовать ли алгоритмы определения количества людей в группе

Работает только при включённом обнаружении типа объекта: человека и машины

4.8.4.7 bool SVmdaParams4::determineNoise

Использовать ли алгоритмы отсечения псевдодвижущихся объектов

4.8.4.8 unsigned int SVmdaParams4::filterStrength

Число кадров [10...50] для фильтрации движения.

Чем больше число этих кадров – тем плавнее будет движение объекта.

4.8.4.9 CFARegionOfInterest SVmdaParams4::roi

Интересующая нас (в плане трекинга и оставленных предметов) область поля зрения

4.8.4.10 bool SVmdaParams4::singleRegion

Использовать (true) или нет (false) 3D измерения по размеченной сцене

4.8.4.11 bool SVmdaParams4::skipEqualFrames

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

4.8.4.12 double SVmdaParams4::timeUntilLost

Время в секундах до удаления пропавшего объекта из базы

4.8.4.13 CFASizeConstraint SVmdaParams4::trackerHeightConstraint

Ограничения на высоту движущегося объекта

4.8.4.14 double SVmdaParams4::trackerSensitivity

Чувствительность детектора движения [0...100].

Если равно 0, то датчик движения использует алгоритм автоматического определения чувствительности.

4.8.4.15 CFASizeConstraint SVmdaParams4::trackerWidthConstraint

Ограничения на ширину движущегося объекта

4.8.4.16 CFASizeConstraint SVmdaParams4::unattendedObjectHeightConstraint

Ограничения на высоту появившегося/исчезнувшего предмета

4.8.4.17 double SVmdaParams4::unattendedObjectSensitivity

Чувствительность детектора появившихся/исчезнувших предметов [1...25].

4.8.4.18 CFASizeConstraint SVmdaParams4::unattendedObjectWidthConstraint

Ограничения на ширину появившегося/исчезнувшего предмета

4.8.4.19 int SVmdaParams4::useAntiShaker

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

4.8.4.20 bool SVmdaParams4::useCalibrationData

Использовать ли перспективные измерения

Объявления и описания членов структуры находятся в файле:

- VMDAParams4.h

4.9 Структура VMDA_IRecognizer

Интерфейс на стороне библиотеки анализа движения, вызываемый клиентским приложением

#include <iit-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 444.
- virtual void OnFrame_YUV420 (unsigned int nStrideY, unsigned int nStrideUV, 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 420.
- virtual void OnFrame_YUV422 (unsigned int nStrideY, unsigned int nStrideUV, 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 422.
- virtual void Flush ()=0

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

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

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

  Замена параметров детектирования, не останавливая процесс прослеживания

4.9.1 Подробное описание

Интерфейс на стороне библиотеки анализа движения, вызываемый клиентским приложением

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

4.9.2 Методы

4.9.2.1 virtual void VMDA_IRecognizer::Destroy ( ) [pure virtual]

Удаление объекта, реализующего интерфейс детектора

4.9.2.2 virtual void VMDA_IRecognizer::Flush ( ) [pure virtual]

Команда анализатору на завершение записи всех объектов.

Для каждого объекта, который был создан с помощью CreateObject анализатор вызывает либо CommitObject, либо UndoObject

4.9.2.3 virtual void VMDA_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 444.

Аргументы

nStride	величина страйда поступившего кадра
dtPosixTime	момент времени поступления кадра, указывается как double-значение от произвольной точки отсчета
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

4.9.2.4 virtual void VMDA_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, указывается как double-значение от произвольной точки отсчета
frameData	указатель на массив байт яркостей текущего кадра

4.9.2.5 virtual void VMDA_IRecognizer::OnFrame_YUV420 (unsigned int nStrideY, unsigned int nStrideUV, 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 420.

Аргументы

nStrideY	величина страйда для Y-компоненты
nStrideUV	величина страйда для U и V-компонент
dtPosixTime	момент времени поступления кадра, указывается как double-значение от произвольной точки отсчета

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

4.9.2.6 virtual void VMDA_IRecognizer::OnFrame_YUV422 (unsigned int nStrideY, unsigned int nStrideUV, 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 422.

Аргументы

nStrideY	величина страйда для Y-компоненты
nStrideUV	величина страйда для U и V-компонент
dtPosixTime	момент времени поступления кадра, указывается как double-значение от произвольной точки отсчета
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

4.9.2.7 virtual void VMDA_IRecognizer::Renew ( ) [pure virtual]

Очистка внутренней информации библиотеки обо всех имеющихся объектах.

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

4.9.2.8 virtual void VMDA_IRecognizer::SwapParams (const SVmdaParams * pParams) [pure virtual]

Замена параметров детектирования, не останавливая процесс прослеживания

Меняет параметры детектора, кроме detectUnattendedObjects, useAntiShaker, filterStrength

Аргументы

pParams	указатель на структуру с новыми параметрами детектора

Объявления и описания членов структуры находятся в файле:

- iit-detector.h

4.10 Структура VMDA_IResultsCollector

Интерфейс на стороне клиентского приложения, вызываемый библиотекой анализа движения

#include <iit-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 объекта с данным ID на текущем кадре.
- virtual void SetGeometry3D (int objectId, double x, double y, double height)=0

  Установка пространственных координат и высоты объекта с данным ID на текущем кадре.
- virtual void SetType (int objectId, VMDA_EObjType ObjType, int typeValue)=0

Установка типа объекта с данным ID на текущем кадре

- virtual void CommitObject (int objectId)=0

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

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

  Проверка заполненности коллектора объектов.

4.10.1 Подробное описание

Интерфейс на стороне клиентского приложения, вызываемый библиотекой анализа движения

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

4.10.2 Методы

4.10.2.1 virtual void VMDA_IResultsCollector::CommitObject (int objectId) [pure virtual]

Удаление объекта с данным ID с записью трека в базу.

Т.е. данная процедура сообщает приложению, что объект с данным ID закончил движение по изображению и больше отслеживаться не будет.

Аргументы

objectId	ID удаляемого объекта

4.10.2.2 virtual int VMDA_IResultsCollector::CreateObject ( ) [pure virtual]

Создание нового объекта и начало записи его трека.

Возвращает

ID номер нового объекта

4.10.2.3 virtual bool VMDA_IResultsCollector::IsFull ( ) const [pure virtual]

Проверка заполненности коллектора объектов.

Возвращает

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

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

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

Передача координат объекта с ID объекта, равным objectId. Если эта функция не вызывалась несколько кадров и объект не был удалён из базы, считается, что объект не изменял своих координат на этих кадрах.

Аргументы

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

4.10.2.5 virtual void VMDA_IResultsCollector::SetGeometry3D (int objectId, double x, double y, double height) [pure virtual]

Установка пространственных координат и высоты объекта с данным ID на текущем кадре.

Процедура вызывается как дополнение к процедуре SetGeometry, указывая метрические координаты объекта на сцене

Аргументы

objectId	ID объекта, для которого устанавливаются координаты и высота
x	пространственная X-координата центра основания объекта на изображении
y	пространственная Y-координата центра основания объекта на изображении
height	пространственная высота объекта на изображении

4.10.2.6 virtual void VMDA_IResultsCollector::SetType (int objectId, VMDA_EObjType ObjType, int typeValue) [pure virtual]

Установка типа объекта с данным ID на текущем кадре

Аргументы

objectId	ID объекта, для которого устанавливается тип
ObjType	тип объекта, задаваемый перечислением VMDA_EObjType
typeValue	используется только для типа "Человек" (ObjType=1) и определяет количество людей в группе

4.10.2.7 virtual void VMDA_IResultsCollector::UndoObject (int objectId) [pure virtual]

Удаление объекта с данным ID с удалением трека из базы.

Т.е. данная процедура сообщает приложению, что объект с данным ID - на самом деле артефакт.

Аргументы

objectId	ID удаляемого объекта

Объявления и описания членов структуры находятся в файле:

- iit-detector.h

5 Файлы

5.1 Файл iit-detector.h

Классы

- struct SVmdaParams

  Базовый класс параметров детектора
- struct VMDA_IResultsCollector

  Интерфейс на стороне клиентского приложения, вызываемый библиотекой анализа движения
- struct VMDA_IRecognizer

  Интерфейс на стороне библиотеки анализа движения, вызываемый клиентским приложением

Макросы

- #define IITEXPORT extern "C"

Перечисления

- enum VMDA_EObjType {
  unknown = -1, other = 0, human = 1, car = 2,
  noise = 3, given = 4, taken = 5 }

  перечисление типов объектов (машина, человек, ...)

Функции

- asm (".section .drectve")

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

  функция создания библиотеки в ОС Linux.
- IITEXPORT VMDA_IRecognizer * VMDA_CreateRecognizer (int nFrameWidth, int nFrameHeight, VMDA_IResultsCollector *pMDCollector, VMDA_IResultsCollector *pNOMODCollector, const SVmdaParams *pParams, char *szErrDescBuf, int nErrDescBufLength)

5.1.1 Макросы

5.1.1.1 #define IITEXPORT extern "C"

5.1.2 Перечисления

5.1.2.1 enum VMDA_EObjType

перечисление типов объектов (машина, человек, ...)

Элементы перечислений:

unknown неизвестный

other прочее

human человек

car машина

noise шум

given принесённый предмет (используется в режиме детектирования оставленных предметов)

taken унесённый предмет (используется в режиме детектирования оставленных предметов)

5.1.3 Функции

5.1.3.1 asm (".section .drectve")

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

5.1.3.2 asm (".ascii \"-export:VMDA_CreateRecognizer\"")

функция создания библиотеки в ОС Linux.

5.1.3.3 IITEXPORT VMDA_IRecognizer* VMDA_CreateRecognizer (int nFrameWidth, int nFrameHeight, VMDA_IResultsCollector * pMDCollector, VMDA_IResultsCollector * pNOMODCollector, const SVmdaParams * pParams, char * szErrDescBuf, int nErrDescBufLength)

функция, создающая библиотеку с параметрами pParams

Возвращает

указатель на экземпляр класса VMDA_IRecognizer

Аргументы

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

5.2 Файл VMDAParams4.h

#include "iit-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 CFACalibrationData

  Данные калибровки камеры и разметки сцены, необходимые для измерения перспективы
- struct CFASizeConstraint

  Ограничения на размер детектируемого объекта
- struct CFARegionOfInterest

  Области интереса и области игнорирования кадра.
- struct SVmdaParams4

  Основной класс параметров текущей версии детектора

Макросы

- #define _VMDA_PARAMS4_IIT_H_

5.2.1 Макросы

5.2.1.1 #define _VMDA_PARAMS4_IIT_H_

6 Примеры работы с библиотекой

6.1 Пример работы с объектом

Примерный сценарий добавления объекта в коллектор:

```
 \{пришел кадр №1, объект в точке (0.1; 0.1)\}
\textcolor{keywordtype}{int} objectId = pCollector->CreateObject(); \(\backslash\)n
pCollector->SetGeometry(objectId, 0.1, 0.1, 0.01, 0.01);


 \{пришел кадр №2, объект перешел в точку (0.2; 0.2)\}
pCollector->SetGeometry(objectId, 0.2, 0.2, 0.01, 0.01);


 \{пришел кадр №3, объект перешел в точку (0.4; 0.2)\}
pCollector->SetGeometry(objectId, 0.4, 0.2, 0.01, 0.01);


 \{пришел кадр №4, объект исчез\}
\textcolor{keywordflow}{if}(объект является ложным)
      pCollector->UndoObject(objectId);
\textcolor{keywordflow}{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.

• Отличие библиотеки версии 9.x от прошлых версий заключается в том, что параметр Stride из функции CreateRecognizer перенесен в функции OnFrame8bit и OnFrame24bit_YUV. Stride – ширина выровненного кадра (возможно, с дополнительной информацией) в пикселях. В данном случае при обработке изображения вертикальная полоса ширины stride-width не учитывается и измерения нормируются по действительной ширине кадра (см. рисунок)

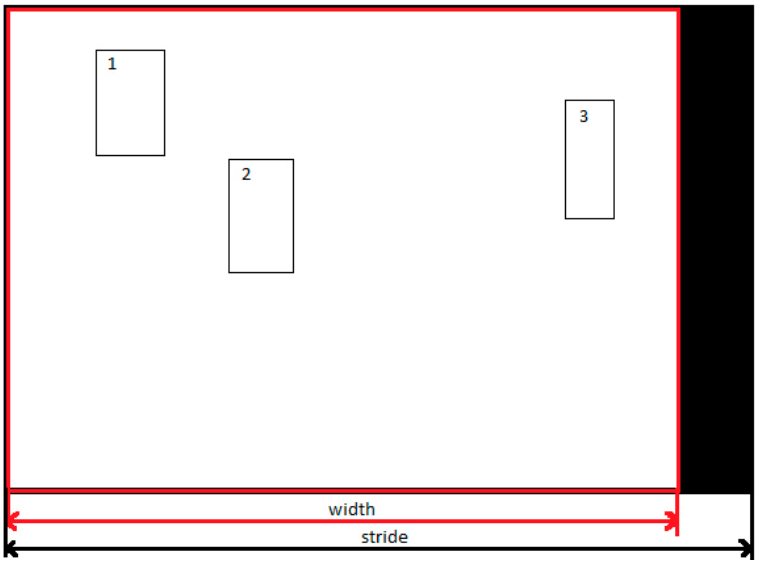

Рис. 1: Графический пример соотношения параметров stride и width

• Минимальный размер объекта, который детектируется алгоритмом, равен 36 квадратным пикселям.

• Минимальная скорость детектируемого объекта 0.5-1 пиксель/кадр (зависит от размеров объекта).

• Области интереса используются для выделения областей слежения от областей с шумами (деревья, отражения, волны). Области игнорирования используются для выделения шумов из больших областей интереса. При том чем меньше областей интереса, тем быстрее работает библиотека.

• Если trackerSensitivity=0, используется алгоритм автоматического выбора чувствительности.

• Примерная разметка кадров при использовании перспективы useCalibrationData=true показана на рисунке.

Рис. 2: Разметка кадра с измерением перспективы

• unattendedObjectSizeConstraint следует аккуратно использовать для определения минимальных

и максимальных размеров принесённых или оставленных объектов так, чтобы алгоритм NOM-OD не выделял такие объекты, как остановившиеся люди, вставшие/севшие люди, автомобили, качающиеся на одном месте ветки деревьев.

Предметный указатель