Challenges and Benchmarks

Evaluation of Background Subtraction Algorithms

Benjamin Laugraud and Thierry Bouwmans

Montefiore Institute, University of Liège, Belgium
MIA Laboratory, University of La Rochelle, France

August 28th, 2018

IAPR Summer School on Machine and Vision Intelligence (VISMAC 2018)

Vico Equense, Naples, Italy

Purpose

- Modern background subtraction (BGS) algorithms must be evaluated rigorously!
- Such an evaluation should consider all the challenges associated with the field.
- Important to convince reviewers and readers of the efficacy of a method.
- How to perform a rigorous evaluation in practice?

Important Topics to Study

ChangeDetection.NET (CDnet) Dataset

- Content of the website.
- Structure and conventions of the dataset.
- Evaluation tools associated with the dataset.

BGSLibrary

- Presentation.
- Content of the website.
- Structure and conventions of the library.

C++ Programming

- How to use an algorithm from the BGSLibrary in your own C++ code?
- How to apply an algorithm from the BGSLibrary on CDnet?
- How to integrate your own algorithm in the BGSLibrary?

Prerequisites

- The **Ubuntu** (or derived) GNU/Linux distribution.[1]
- The **OpenCV** library.[2]
- The **CMake** compilation utility.[3]
- The **Matlab** programming environment.[4]

[1] https://www.ubuntu.com

[2] https://opencv.org

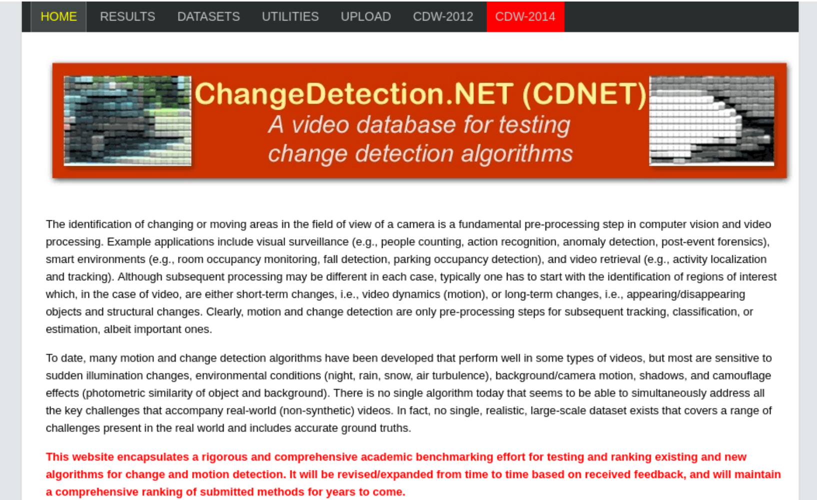
[3] https://cmake.org

[4] https://www.mathworks.com/products/matlab.html

CDnet Dataset [2] [4]

Website

http://changedetection.net



ChangeDetection.NET (CDNET)

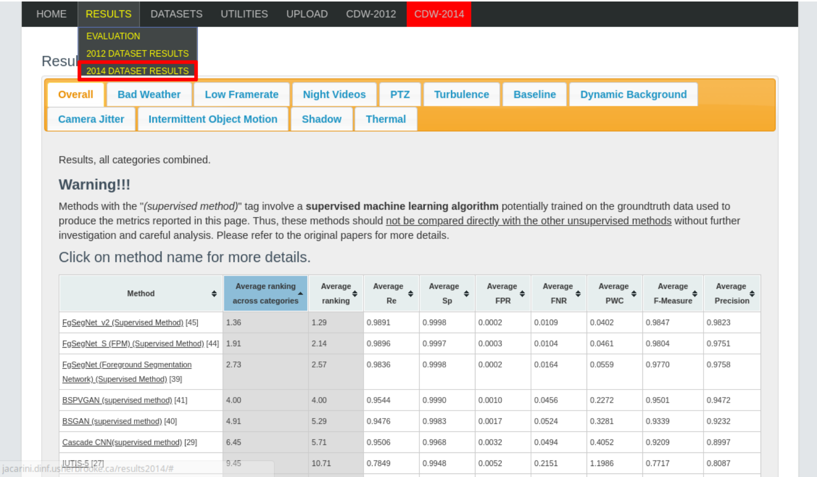
A video database for testing change detection algorithms

The identification of changing or moving areas in the field of view of a camera is a fundamental pre-processing step in computer vision and video processing. Example applications include visual surveillance (e.g., people counting, action recognition, anomaly detection, post-event forensics), smart environments (e.g., room occupancy monitoring, fall detection, parking occupancy detection), and video retrieval (e.g., activity localization and tracking). Although subsequent processing may be different in each case, typically one has to start with the identification of regions of interest which, in the case of video, are either short-term changes, i.e., video dynamics (motion), or long-term changes, i.e., appearing/disappearing objects and structural changes. Clearly, motion and change detection are only pre-processing steps for subsequent tracking, classification, or estimation, albeit important ones.

To date, many motion and change detection algorithms have been developed that perform well in some types of videos, but most are sensitive to sudden illumination changes, environmental conditions (night, rain, snow, air turbulence), background/camera motion, shadows, and camouflage effects (photometric similarity of object and background). There is no single algorithm today that seems to be able to simultaneously address all the key challenges that accompany real-world (non-synthetic) videos. In fact, no single, realistic, large-scale dataset exists that covers a range of challenges present in the real world and includes accurate ground truths.

This website encapsulates a rigorous and comprehensive academic benchmarking effort for testing and ranking existing and new algorithms for change and motion detection. It will be revised/expanded from time to time based on received feedback, and will maintain a comprehensive ranking of submitted methods for years to come.

Overall Results



Results, all categories combined.

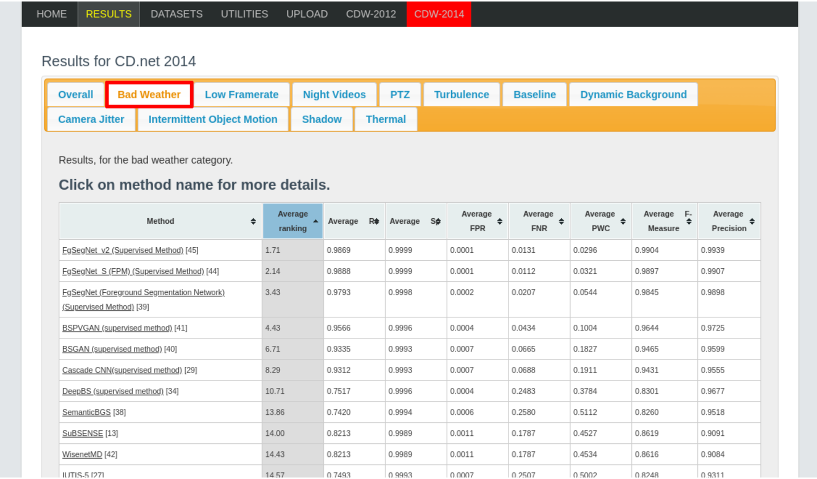
Warning!!!

Methods with the "*(supervised method)*" tag involve a **supervised machine learning algorithm** potentially trained on the groundtruth data used to produce the metrics reported in this page. Thus, these methods should not be compared directly with the other unsupervised methods without further investigation and careful analysis. Please refer to the original papers for more details.

Click on method name for more details.

Method	Average ranking across categories	Average ranking	Average Re	Average Sp	Average FPR	Average FNR	Average PWC	Average F-Measure	Average Precision
FgSegNet_v2 (Supervised Method) [45]	1.36	1.29	0.9891	0.9998	0.0002	0.0109	0.0402	0.9847	0.9823
FgSegNet_S (FPM) (Supervised Method) [44]	1.91	2.14	0.9896	0.9997	0.0003	0.0104	0.0461	0.9804	0.9751
FgSegNet (Foreground Segmentation Network) (Supervised Method) [39]	2.73	2.57	0.9836	0.9998	0.0002	0.0164	0.0559	0.9770	0.9758
BSPVGAN (supervised method) [41]	4.00	4.00	0.9544	0.9990	0.0010	0.0456	0.2272	0.9501	0.9472
BSGAN (supervised method) [40]	4.91	5.29	0.9476	0.9983	0.0017	0.0524	0.3281	0.9339	0.9232
Cascade CNN(supervised method) [29]	6.45	5.71	0.9506	0.9968	0.0032	0.0494	0.4052	0.9209	0.8997
IUTIS-5 [27]	9.45	10.71	0.7849	0.9948	0.0052	0.2151	1.1986	0.7717	0.8087

Category Results (Bad Weather for the Example)



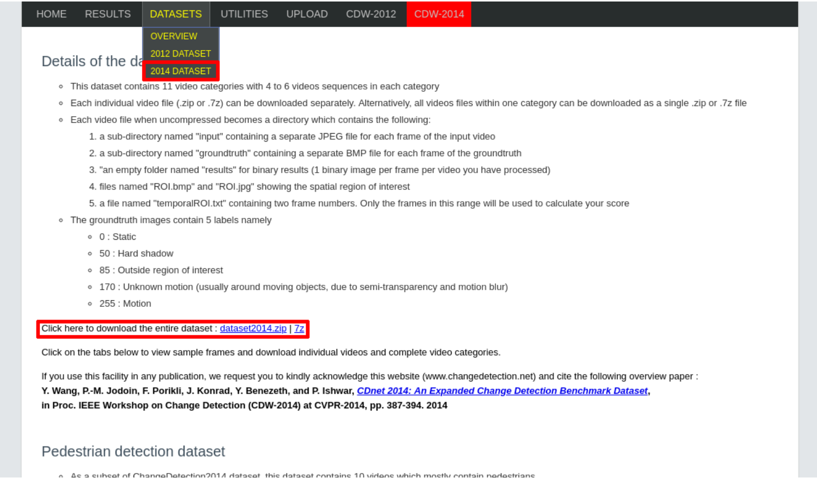
Results for CD.net 2014

Results, for the bad weather category.

Click on method name for more details.

Method	Average ranking	Average R	Average S	Average FPR	Average FNR	Average PWC	Average F-Measure	Average Precision
FgSegNet_v2 (Supervised Method) [45]	1.71	0.9869	0.9999	0.0001	0.0131	0.0296	0.9904	0.9939
FgSegNet_S (FPM) (Supervised Method) [44]	2.14	0.9888	0.9999	0.0001	0.0112	0.0321	0.9897	0.9907
FgSegNet (Foreground Segmentation Network) (Supervised Method) [39]	3.43	0.9793	0.9998	0.0002	0.0207	0.0544	0.9845	0.9898
BSPVGAN (supervised method) [41]	4.43	0.9566	0.9996	0.0004	0.0434	0.1004	0.9644	0.9725
BSGAN (supervised method) [40]	6.71	0.9335	0.9993	0.0007	0.0665	0.1827	0.9465	0.9599
Cascade CNN(supervised method) [29]	8.29	0.9312	0.9993	0.0007	0.0688	0.1911	0.9431	0.9555
DeepBS (supervised method) [34]	10.71	0.7517	0.9996	0.0004	0.2483	0.3784	0.8301	0.9677
SemanticBGS [38]	13.86	0.7420	0.9994	0.0006	0.2580	0.5112	0.8260	0.9518
SuBSENSE [13]	14.00	0.8213	0.9989	0.0011	0.1787	0.4527	0.8619	0.9091
WisenetMD [42]	14.43	0.8213	0.9989	0.0011	0.1787	0.4534	0.8616	0.9084
IUTIS-5 [27]	14.57	0.7493	0.9993	0.0007	0.2507	0.5002	0.8248	0.9311

Download



Details of the da

- This dataset contains 11 video categories with 4 to 6 videos sequences in each category
- Each individual video file (.zip or .7z) can be downloaded separately. Alternatively, all videos files within one category can be downloaded as a single .zip or .7z file
- Each video file when uncompressed becomes a directory which contains the following:
  1. a sub-directory named "input" containing a separate JPEG file for each frame of the input video
  2. a sub-directory named "groundtruth" containing a separate BMP file for each frame of the groundtruth
  3. "an empty folder named "results" for binary results (1 binary image per frame per video you have processed)
  4. files named "ROI.bmp" and "ROI.jpg" showing the spatial region of interest
  5. a file named "temporalROI.txt" containing two frame numbers. Only the frames in this range will be used to calculate your score
- The groundtruth images contain 5 labels namely
  - 0 : Static
  - 50 : Hard shadow
  - 85 : Outside region of interest
  - 170 : Unknown motion (usually around moving objects, due to semi-transparency and motion blur)
  - 255 : Motion

Click here to download the entire dataset : dataset2014.zip | 7z

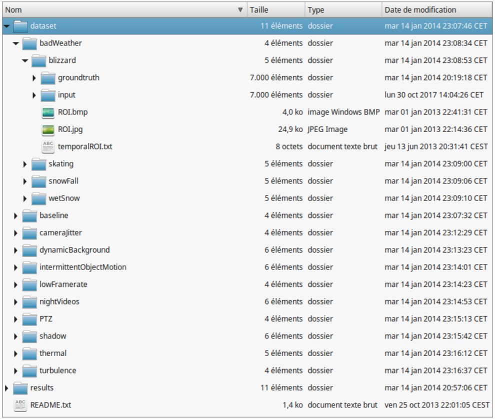
Click on the tabs below to view sample frames and download individual videos and complete video categories.

If you use this facility in any publication, we request you to kindly acknowledge this website (www.changedetection.net) and cite the following overview paper :
Y. Wang, P.-M. Jodoin, F. Porikli, J. Konrad, Y. Benezeth, and P. Ishwar, *CDnet 2014: An Expanded Change Detection Benchmark Dataset*,
in Proc. IEEE Workshop on Change Detection (CDW-2014) at CVPR-2014, pp. 387-394. 2014

Pedestrian detection dataset

- As a subset of ChangeDetection2014 dataset, this dataset contains 10 videos which mostly contain pedestrians

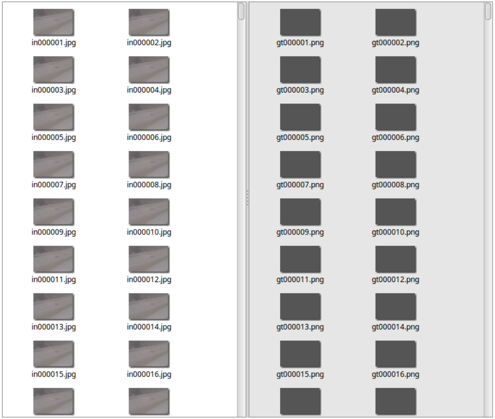
Dataset Structure



Nom	Taille	Type	Date de modification
dataset	11 éléments	dossier	mar 14 jan 2014 23:07:46 CET
badWeather	4 éléments	dossier	mar 14 jan 2014 23:08:34 CET
blizzard	5 éléments	dossier	mar 14 jan 2014 23:08:53 CET
groundtruth	7.000 éléments	dossier	mar 14 jan 2014 20:19:18 CET
input	7.000 éléments	dossier	lun 30 oct 2017 14:04:26 CET
ROI.bmp	4,0 ko	image Windows BMP	mar 01 jan 2013 22:41:31 CET
ROI.jpg	24,9 ko	JPEG Image	mar 01 jan 2013 22:14:36 CET
temporalROI.txt	8 octets	document texte brut	jeu 13 jun 2013 20:31:41 CEST
skating	5 éléments	dossier	mar 14 jan 2014 23:09:00 CET
snowFall	5 éléments	dossier	mar 14 jan 2014 23:09:06 CET
wetSnow	5 éléments	dossier	mar 14 jan 2014 23:09:10 CET
baseline	4 éléments	dossier	mar 14 jan 2014 23:07:32 CET
cameraJitter	4 éléments	dossier	mar 14 jan 2014 23:12:29 CET
dynamicBackground	6 éléments	dossier	mar 14 jan 2014 23:13:23 CET
intermittentObjectMotion	6 éléments	dossier	mar 14 jan 2014 23:14:01 CET
lowFramerate	4 éléments	dossier	mar 14 jan 2014 23:14:23 CET
nightVideos	6 éléments	dossier	mar 14 jan 2014 23:14:53 CET
PTZ	4 éléments	dossier	mar 14 jan 2014 23:15:13 CET
shadow	6 éléments	dossier	mar 14 jan 2014 23:15:42 CET
thermal	5 éléments	dossier	mar 14 jan 2014 23:16:12 CET
turbulence	4 éléments	dossier	mar 14 jan 2014 23:16:37 CET
results	11 éléments	dossier	mar 14 jan 2014 20:57:06 CET
README.txt	1,4 ko	document texte brut	ven 25 oct 2013 22:01:05 CEST

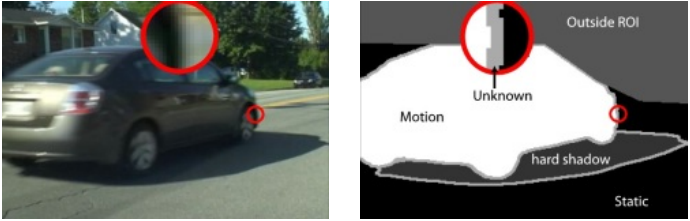
- Once uncompressed, the dataset is in a `dataset` folder.
- Inside, there is a folder per category gathering a folder per sequence.
- For each sequence, there is a folder for the `input` and the `groundtruth`.
- In `temporalROI.txt` there are 2 frame numbers defining the evaluation interval.

Sequence Content



- The `input` folder of a given sequence contains one `.jpg` file per frame.
- The name of a file is `in`, the frame number encoded with 6 digits, and `.jpg`.
- The `groundtruth` folder contains one `.png` file per frame.
- The name of a file is `gt`, the frame number encoded with 6 digits, and `.png`.

Ground-Truth Format



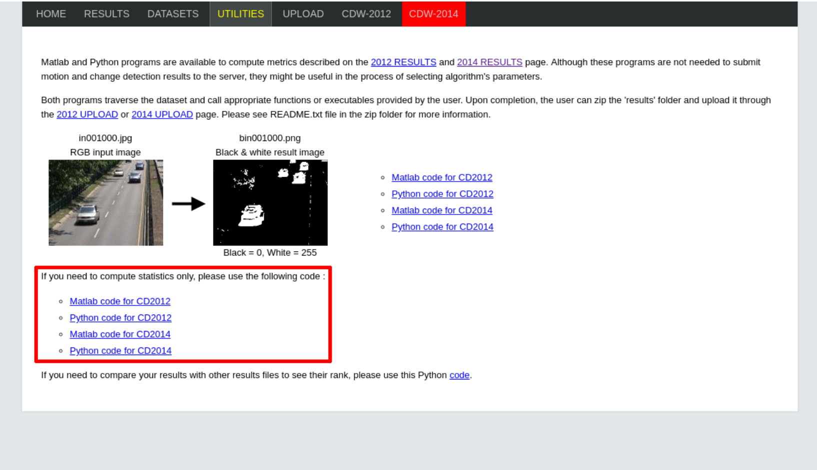
(Taken from the CDnet website)

- Each pixel of a ground-truth (GT) `.png` file has a value among:

0	Background
50	Shadow
85	Outside the ROI (pixel ignored during the evaluation)
170	Impossible to determine (pixel ignored during the evaluation)
255	Foreground

- Note: A frame outside the evaluation interval has a GT full of `85` values.

Tools for Performance Evaluation



Matlab and Python programs are available to compute metrics described on the 2012 RESULTS and 2014 RESULTS page. Although these programs are not needed to submit motion and change detection results to the server, they might be useful in the process of selecting algorithm's parameters.

Both programs traverse the dataset and call appropriate functions or executables provided by the user. Upon completion, the user can zip the 'results' folder and upload it through the 2012 UPLOAD or 2014 UPLOAD page. Please see README.txt file in the zip folder for more information.

in001000.jpg
RGB input image

bin001000.png
Black & white result image

Black = 0, White = 255

- Matlab code for CD2012
- Python code for CD2012
- Matlab code for CD2014
- Python code for CD2014

If you need to compute statistics only, please use the following code :

- Matlab code for CD2012
- Python code for CD2012
- Matlab code for CD2014
- Python code for CD2014

If you need to compare your results with other results files to see their rank, please use this Python code.

- Some tools to compute metrics (e.g. F1) are given along with the dataset.
- There are Matlab and Python versions.
- In this tutorial, we will show how to use the Matlab version.

Small problem with GNU/Linux

- To work on GNU/Linux, the CDnet evaluation tool requires some modifications.
- In `processVideoFolder.m` line `35`:

```
fID = fopen([path, ’\temporalROI.txt’]); % Before
fID = fopen([path, ’/temporalROI.txt’]); % After
```

- In `Stats.m` line `45`:

```
% Before
f = fopen([this.path ’\’ category ’\’ video ’\cm.txt’], ’wt’);
% After
f = fopen([this.path ’/’ category ’/’ video ’/cm.txt’], ’wt’);
```

- In `Stats.m` line `52`:

```
f = fopen([this.path ’\’ category ’\cm.txt’], ’wt’); % Before
f = fopen([this.path ’/’ category ’/cm.txt’], ’wt’); % After
```

- In `Stats.m` line `76`:

```
f = fopen([this.path ’\cm.txt’], ’wt’); % Before
f = fopen([this.path ’/cm.txt’], ’wt’); % After
```

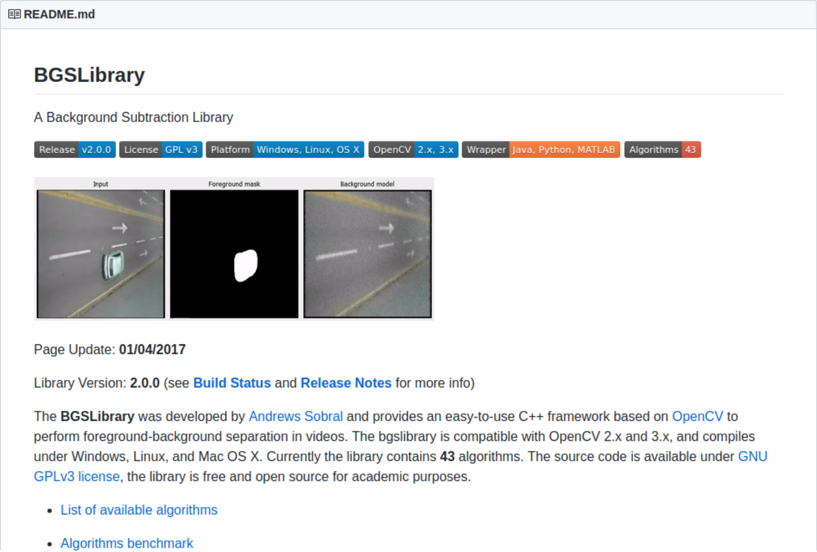
BGSLibrary [3] [1]

Presentation

- Open-source (GPL 3) C++ library full of BGS algorithms.
- Based upon OpenCV.
- Maintained by Andrews Sobral.
- Numerous algorithms have been implemented by the authors!
- Provides also: GUI; wrappers for Java, Python, and Matlab; Docker images; etc.
- Everyone is free to send its algorithm (as long as a reference is associated).
- For any support related to the BGSLibrary, please contact Andrews Sobral.

Website

https://github.com/andrewssobral/bgslibrary



README.md

BGSLibrary

A Background Subtraction Library

Release v2.0.0 License GPL v3 Platform Windows, Linux, OS X OpenCV 2.x, 3.x Wrapper Java, Python, MATLAB Algorithms 43

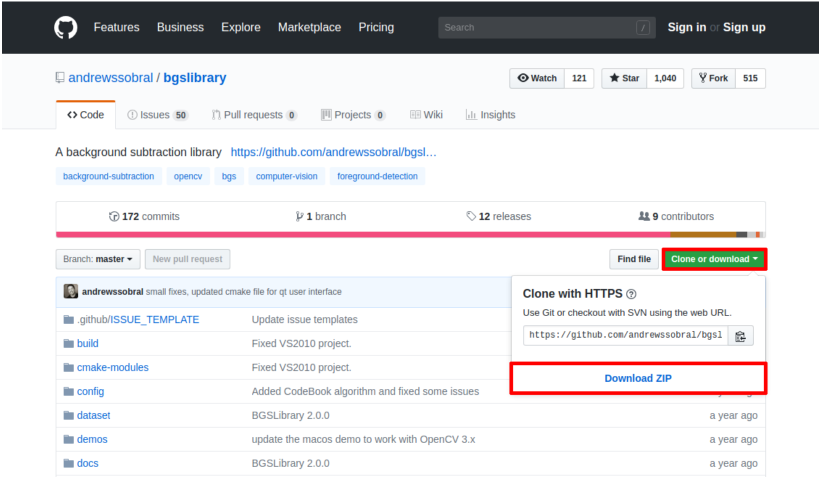
Page Update: **01/04/2017**

Library Version: **2.0.0** (see **Build Status** and **Release Notes** for more info)

The **BGSLibrary** was developed by Andrews Sobral and provides an easy-to-use C++ framework based on OpenCV to perform foreground-background separation in videos. The bgslibrary is compatible with OpenCV 2.x and 3.x, and compiles under Windows, Linux, and Mac OS X. Currently the library contains **43** algorithms. The source code is available under GNU GPLv3 license, the library is free and open source for academic purposes.

- List of available algorithms
- Algorithms benchmark

Download



andrewssobral / **bgslibrary**

A background subtraction library https://github.com/andrewssobral/bgsl…

background-subtraction · opencv · bgs · computer-vision · foreground-detection

172 commits · **1** branch · **12** releases · **9** contributors

andrewssobral small fixes, updated cmake file for qt user interface

Clone with HTTPS

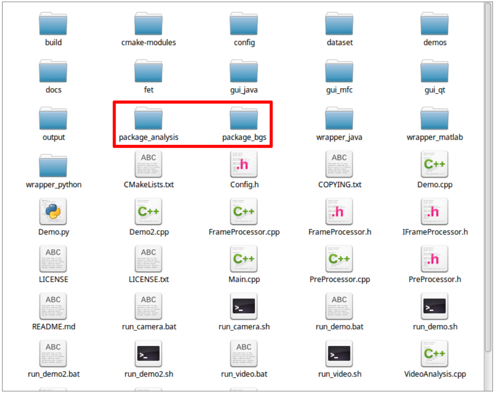
Use Git or checkout with SVN using the web URL.

`https://github.com/andrewssobral/bgsl`

Download ZIP

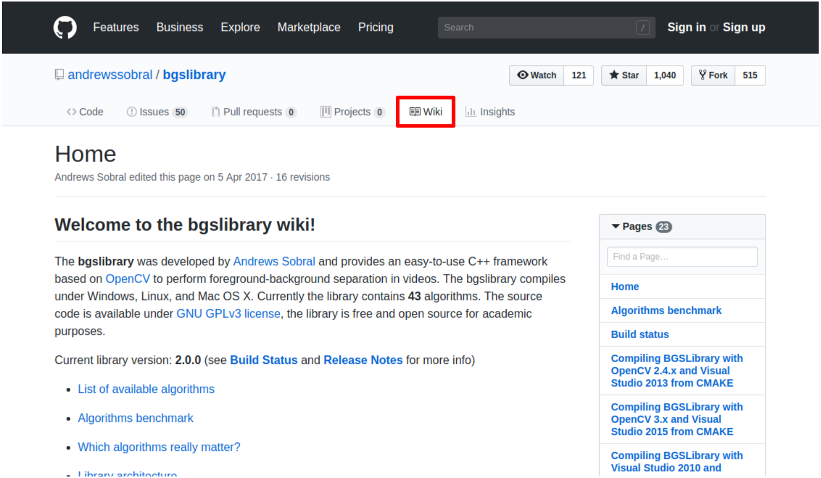
.github/ISSUE_TEMPLATE	Update issue templates	
build	Fixed VS2010 project.	
cmake-modules	Fixed VS2010 project.	
config	Added CodeBook algorithm and fixed some issues	
dataset	BGSLibrary 2.0.0	a year ago
demos	update the macos demo to work with OpenCV 3.x	a year ago
docs	BGSLibrary 2.0.0	a year ago

Library Structure



- Once uncompressed, you have all the files to compile the library and its tools.
- There are Java and QT GUIs, but...
- ...in this tutorial, we will focus on the inclusion of the library in your own program.
- This requires to copy in your project `package_analysis` and `package_bgs`.
- The folder `package_bgs` contains the implementations of the BGS algorithms.

Wiki


Features
Business
Explore
Marketplace
Pricing
Search
Sign in or Sign up
andrewssobral / bgslibrary
Watch 121
Star 1,040
Fork 515
Code
Issues 50
Pull requests 0
Projects 0
Wiki
Insights
Pages 23
Find a Page…
Home
Algorithms benchmark
Build status
Compiling BGSLibrary with OpenCV 2.4.x and Visual Studio 2013 from CMAKE
Compiling BGSLibrary with OpenCV 3.x and Visual Studio 2015 from CMAKE
Compiling BGSLibrary with Visual Studio 2010 and

Home

Andrews Sobral edited this page on 5 Apr 2017 · 16 revisions

Welcome to the bgslibrary wiki!

The **bgslibrary** was developed by Andrews Sobral and provides an easy-to-use C++ framework based on OpenCV to perform foreground-background separation in videos. The bgslibrary compiles under Windows, Linux, and Mac OS X. Currently the library contains **43** algorithms. The source code is available under GNU GPLv3 license, the library is free and open source for academic purposes.

Current library version: **2.0.0** (see **Build Status** and **Release Notes** for more info)

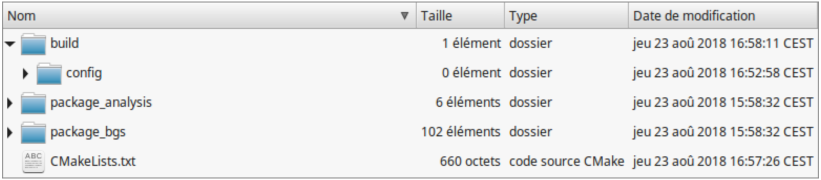
- List of available algorithms
- Algorithms benchmark
- Which algorithms really matter?
- Library architecture

- The wiki on the website documents every aspect of the library.
- You can find the list of available BGS algorithms.

C++ Programming

Step 1: C++ Project with the BGSLibrary

Procedure



Nom	Taille	Type	Date de modification
build	1 élément	dossier	jeu 23 aoû 2018 16:58:11 CEST
config	0 élément	dossier	jeu 23 aoû 2018 16:52:58 CEST
package_analysis	6 éléments	dossier	jeu 23 aoû 2018 15:58:32 CEST
package_bgs	102 éléments	dossier	jeu 23 aoû 2018 15:58:32 CEST
CMakeLists.txt	660 octets	code source CMake	jeu 23 aoû 2018 16:57:26 CEST

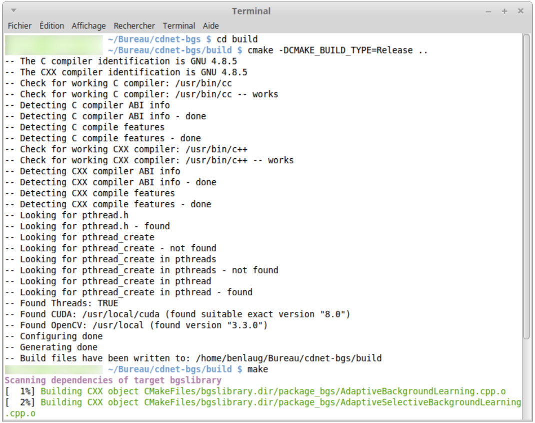
Create a folder for your project and put inside:

- An empty `build` folder (location of the compiled files).
- An empty `build/config` (location of the BGSLibrary config files).
- A copy of the `package_analysis` folder of the BGSLibrary.
- A copy of the `package_bgs` folder of the BGSLibrary.
- An empty `CMakeLists.txt` file.

Minimal CMakeLists.txt File

```
cmake_minimum_required(VERSION 2.8)

# Project name
project(cdnet-bgs)

# Enable C++11 support of the compiler
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11")

# Find OpenCV
find_package(OpenCV REQUIRED)

# Add include directories of OpenCV
include_directories(SYSTEM ${OpenCV_INCLUDE_DIRS})

# Find all C++ and C implementation files in package_bgs
file(GLOB_RECURSE bgslibrary_src
                  package_bgs/*.cpp
                  package_bgs/*.cc
                  package_bgs/*.c)

# Declare the BGSLibrary with the files being found
add_library(bgslibrary STATIC ${bgslibrary_src})

# Link the BGSLibrary with OpenCV
target_link_libraries(bgslibrary ${OpenCV_LIBS})
```

Compilation

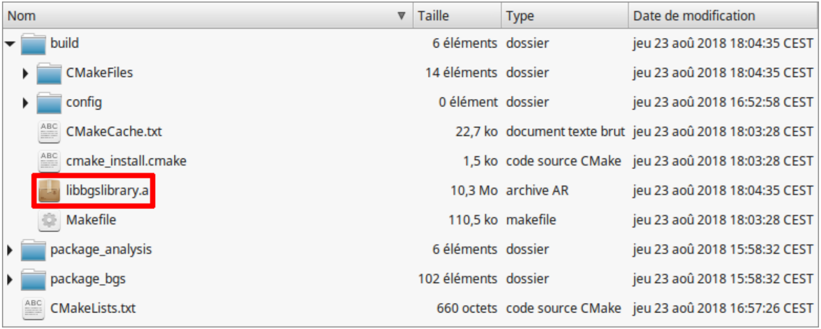


```
[redacted] ~/Bureau/cdnet-bgs $ cd build
[redacted] ~/Bureau/cdnet-bgs/build $ cmake -DCMAKE_BUILD_TYPE=Release ..
-- The C compiler identification is GNU 4.8.5
-- The CXX compiler identification is GNU 4.8.5
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Looking for pthread.h
-- Looking for pthread.h - found
-- Looking for pthread_create
-- Looking for pthread_create - not found
-- Looking for pthread_create in pthreads
-- Looking for pthread_create in pthreads - not found
-- Looking for pthread_create in pthread
-- Looking for pthread_create in pthread - found
-- Found Threads: TRUE
-- Found CUDA: /usr/local/cuda (found suitable exact version "8.0")
-- Found OpenCV: /usr/local (found version "3.3.0")
-- Configuring done
-- Generating done
-- Build files have been written to: /home/benlaug/Bureau/cdnet-bgs/build
[redacted] ~/Bureau/cdnet-bgs/build $ make
Scanning dependencies of target bgslibrary
[  1%] Building CXX object CMakeFiles/bgslibrary.dir/package_bgs/AdaptiveBackgroundLearning.cpp.o
[  2%] Building CXX object CMakeFiles/bgslibrary.dir/package_bgs/AdaptiveSelectiveBackgroundLearning
.cpp.o
```

The empty project can be compiled using the following commands:

- `$ cd build`
- `$ cmake -DCMAKE_BUILD_TYPE=Release ..`
- `$ make`

BGSLibrary Compiled!

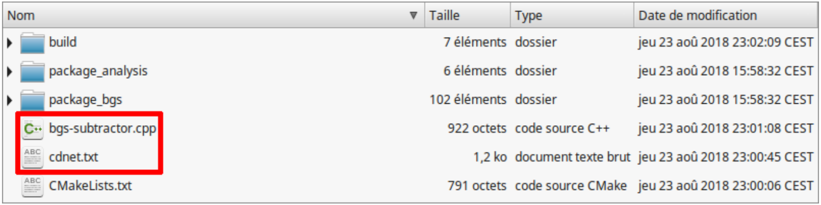


Nom	Taille	Type	Date de modification
build	6 éléments	dossier	jeu 23 aoû 2018 18:04:35 CEST
CMakeFiles	14 éléments	dossier	jeu 23 aoû 2018 18:04:35 CEST
config	0 élément	dossier	jeu 23 aoû 2018 16:52:58 CEST
CMakeCache.txt	22,7 ko	document texte brut	jeu 23 aoû 2018 18:03:28 CEST
cmake_install.cmake	1,5 ko	code source CMake	jeu 23 aoû 2018 18:03:28 CEST
libbgslibrary.a	10,3 Mo	archive AR	jeu 23 aoû 2018 18:04:35 CEST
Makefile	110,5 ko	makefile	jeu 23 aoû 2018 18:03:28 CEST
package_analysis	6 éléments	dossier	jeu 23 aoû 2018 15:58:32 CEST
package_bgs	102 éléments	dossier	jeu 23 aoû 2018 15:58:32 CEST
CMakeLists.txt	660 octets	code source CMake	jeu 23 aoû 2018 16:57:26 CEST

- Once compiled, the BGSLibrary is a static library `libbgslibrary.a` in `build`.
- Thus, we have a project enabling to create a program with the BGSLibrary.

Step 2: List of Sequences

Procedure



Nom	Taille	Type	Date de modification
build	7 éléments	dossier	jeu 23 aoû 2018 23:02:09 CEST
package_analysis	6 éléments	dossier	jeu 23 aoû 2018 15:58:32 CEST
package_bgs	102 éléments	dossier	jeu 23 aoû 2018 15:58:32 CEST
bgs-subtractor.cpp	922 octets	code source C++	jeu 23 aoû 2018 23:01:08 CEST
cdnet.txt	1,2 ko	document texte brut	jeu 23 aoû 2018 23:00:45 CEST
CMakeLists.txt	791 octets	code source CMake	jeu 23 aoû 2018 23:00:06 CEST

- For applying BGS on the CDnet dataset, we need a list of sequences.
- We can add to our project a `cdnet.txt` file containing such a list.
- Thus, we start our program with a function to read `cdnet.txt`.
- The code of the program will be put in a file called `bgs-subtractor.cpp`.

Content of cdnet.txt

```
badWeather/blizzard
badWeather/skating
badWeather/snowFall
badWeather/wetSnow
baseline/highway
baseline/office
baseline/pedestrians
baseline/PETS2006
cameraJitter/badminton
cameraJitter/boulevard
cameraJitter/sidewalk
cameraJitter/traffic
dynamicBackground/boats
dynamicBackground/canoe
dynamicBackground/fall
dynamicBackground/fountain01
dynamicBackground/fountain02
dynamicBackground/overpass
intermittentObjectMotion/abandonedBox
intermittentObjectMotion/parking
intermittentObjectMotion/sofa
intermittentObjectMotion/streetLight
intermittentObjectMotion/tramstop
...
turbulence/turbulence3
```

Modification of CMakeLists.txt

We need to add some lines at the end of CMakeLists.txt to compile our program!

```
...
# Produce the executable bgs-subtractor from the C++ code
add_executable(bgs-subtractor bgs-subtractor.cpp)

# Link bgs-subtractor to the BGSLibrary and OpenCV
target_link_libraries(bgs-subtractor bgslibrary ${OpenCV_LIBS})
```

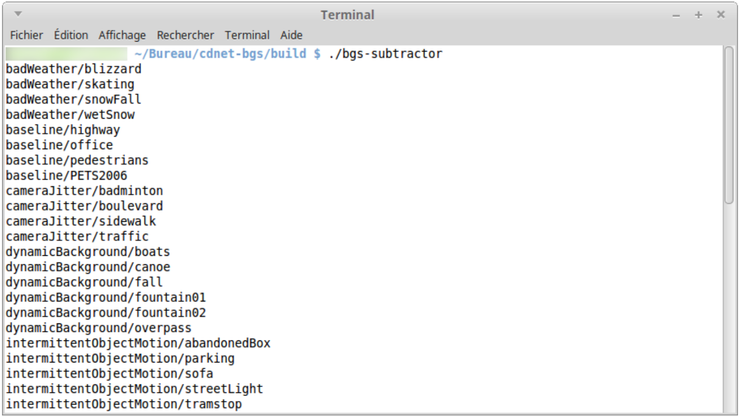
Function `list_seqs()` to read `cdnet.txt` in `bgs-subtractor.cpp`

```cpp
#include <cstddef>
#include <fstream>
#include <iostream>
#include <string>
#include <vector>

using namespace std;

// Path to the cdnet.txt file
#define SEQS_PATH "/home/user/Bureau/cdnet-bgs/cdnet.txt"

// Function to read the cdnet.txt file
vector<string> list_seqs() {
  vector<string> seqs;      // Vector containing the seq. names
  ifstream ifs(SEQS_PATH); // Stream to read the cdnet.txt file
  string buffer;            // String buffer

  // For each line in the cdnet.txt file
  while (getline(ifs, buffer))
    // Add the current sequence name in the vector
    seqs.push_back(buffer);

  // Return the vector of sequence names
  return seqs;
}
```

Code to test `list_seqs()` in `bgs-subtractor.cpp`

```cpp
int main() {
  // Vector with the sequence names
  vector<string> seqs = list_seqs();

  // For each sequence
  for (size_t seq_idx = 0; seq_idx < seqs.size(); ++seq_idx) {
    // Print the current sequence name
    cout << seqs[seq_idx] << endl;
  }
}
```

Compilation and Execution



```
[redacted] ~/Bureau/cdnet-bgs/build $ ./bgs-subtractor
badWeather/blizzard
badWeather/skating
badWeather/snowFall
badWeather/wetSnow
baseline/highway
baseline/office
baseline/pedestrians
baseline/PETS2006
cameraJitter/badminton
cameraJitter/boulevard
cameraJitter/sidewalk
cameraJitter/traffic
dynamicBackground/boats
dynamicBackground/canoe
dynamicBackground/fall
dynamicBackground/fountain01
dynamicBackground/fountain02
dynamicBackground/overpass
intermittentObjectMotion/abandonedBox
intermittentObjectMotion/parking
intermittentObjectMotion/sofa
intermittentObjectMotion/streetLight
intermittentObjectMotion/tramstop
```

- You can compile the code as in the slide 20.
- The program `bgs-subtractor` is in the `build` folder.
- You can launch it with the command `./bgs-subtractor`.
- If everything is correct, the list of CDnet sequences should be printed.

Step 3: Reading a Temporal ROI

Temporal Region of Interest

- In CDnet, each sequence is provided with a `temporalROI.txt` file.
- In contains two integers separated by a space on a unique line.
- The first is the frame number beginning the *evaluation period*.
- The second is the frame number ending the *evaluation period*.
- Our program will save the segmentation maps computed during this period.
- Note that the period before the evaluation period is the *training period*.

Modification of `bgs-subtractor.cpp`

- Add the path to the CDnet dataset.

```
#define  CDNET_PATH "/home/user/Bureau/dataset2014/dataset"
```

- Add a function to read the temporal ROI given a sequence name `seq`.

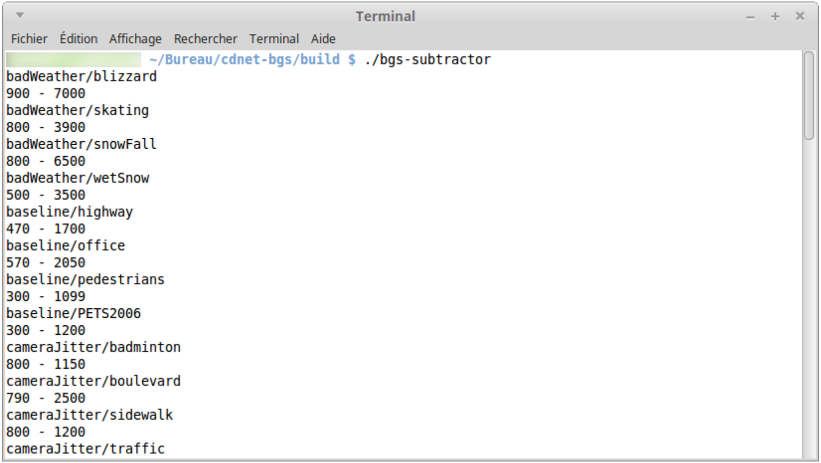
```
vector<int> read_temporal_roi(string seq) {
  /* The temporal ROI is a vector of two integers:
   *   - The first is the frame number beginning the evaluation
   *   - The second is the frame number ending the evaluation
   */
  vector<int> temporal_roi;
  // Stream to read temporalROI.txt
  ifstream ifs(string(CDNET_PATH) + "/" + seq +
               "/temporalROI.txt");
  int frame; // Integer buffer

  ifs >> frame;                    // Read the first integer
  temporal_roi.push_back(frame); // Add it into the vector
  ifs >> frame;                    // Read the second integer
  temporal_roi.push_back(frame); // Add it into the vector

  // Return the temporal ROI vector
  return temporal_roi;
}
```

Code to test `read_temporal_roi()` in `bgs-subtractor.cpp`

For this purpose, we can add the following code in the loop iterating the sequences:

```
...
int main() {
  ...
  // For each sequence
  for (size_t seq_idx = 0; seq_idx < seqs.size(); ++seq_idx) {
    ...
    // Read the temporal ROI of the current sequence
    vector<int> temporal_roi = read_temporal_roi(seqs[seq_idx]);
    // Put the first frame number in a variable frame_begin
    int frame_begin = temporal_roi[0];
    // Put the second frame number in a variable frame_end
    int frame_end   = temporal_roi[1];
    // Print frame_begin and frame_end
    cout << frame_begin << " - " << frame_end << endl;
  }
}
```

Compilation and Execution



```
~/Bureau/cdnet-bgs/build $ ./bgs-subtractor
badWeather/blizzard
900 - 7000
badWeather/skating
800 - 3900
badWeather/snowFall
800 - 6500
badWeather/wetSnow
500 - 3500
baseline/highway
470 - 1700
baseline/office
570 - 2050
baseline/pedestrians
300 - 1099
baseline/PETS2006
300 - 1200
cameraJitter/badminton
800 - 1150
cameraJitter/boulevard
790 - 2500
cameraJitter/sidewalk
800 - 1200
cameraJitter/traffic
```

- You can compile (resp. launch) the code as in the slide 20 (resp. 27).
- If everything is correct, the temporal ROI of each sequence is printed.

Step 4: Reading the Frames of a Sequence

Procedure

- In the sequence folder, we need to read each image file.
- To each image file corresponds a frame.
- For each frame, the image file name has to be formatted correctly.
- We can read the image file using OpenCV.

Modification of `bgs-subtractor.cpp`

- Add the following includes.

```
...
#include <iomanip>
#include <sstream>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
```

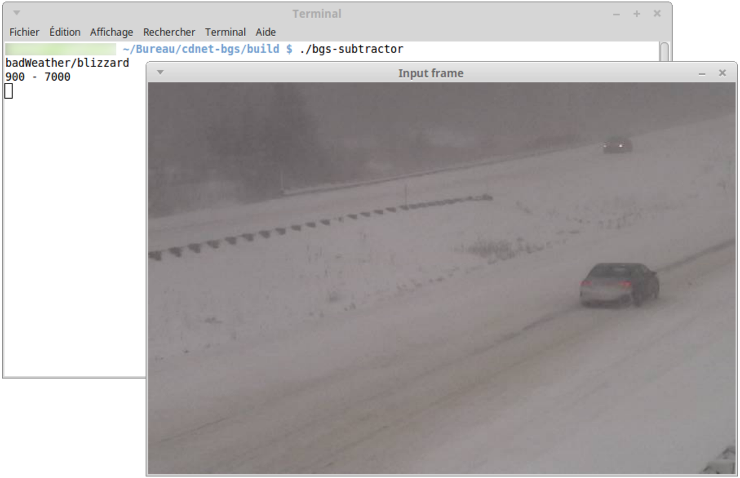
- Use the OpenCV namespace.

```
...
using namespace cv;
```

Code to read and display the frames in `bgs-subtractor.cpp`

```cpp
...
int main() {
  ...
  // For each sequence
  for (size_t seq_idx = 0; seq_idx < seqs.size(); ++seq_idx) {
    ...
    // For each frame
    for (int f_num = 1; f_num <= frame_end; ++f_num) {
      // Stream to format the image file name
      stringstream frame_path;
      // Path to the image file corresponding to the frame
      frame_path << CDNET_PATH << "/" << seqs[seq_idx]
                 << "/input/in" << setw(6) << setfill('0')
                 << f_num << ".jpg";

      // Ask OpenCV to read the image file and put it in a Mat
      Mat frame = imread(frame_path.str());
      // Put the input frame in a graphical window
      imshow("Input frame", frame);
      // Display the graphical window
      waitKey(1);
    }
  }
}
```

Compilation and Execution



```
[redacted] ~/Bureau/cdnet-bgs/build $ ./bgs-subtractor
badWeather/blizzard
900 - 7000
```

- You can compile (resp. launch) the code as in the slide 20 (resp. 27).
- If everything is correct, a window displaying the current sequence appears.

Step 5: Applying a BGS Algorithm

Procedure

- We want to instantiate a given BGS algorithm in the BGSLibrary.
- Apply it on each frame of each CDnet sequence.
- By default, the BGSLibrary displays the results in a graphical window.

Modification of `bgs-subtractor.cpp`

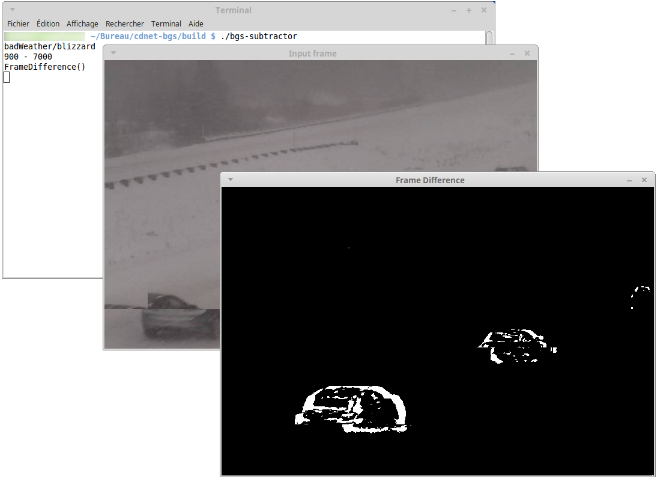
Include the BGSLibrary.

```
...
#include "package_bgs/bgslibrary.h"
```

Code to apply a BGS algorithm in `bgs-subtractor.cpp`

```
...
int main() {
  ...
  // For each sequence
  for (size_t seq_idx = 0; seq_idx < seqs.size(); ++seq_idx) {
    ...
    // Instantiate a BGS algorithm (the frame difference here)
    IBGS* subtractor = new FrameDifference;

    // For each frame
    for (int f_num = 1; f_num <= frame_end; ++f_num) {
      ...
      // Instantiate an empty segmentation map
      Mat seg_map(frame.rows, frame.cols, CV_8UC3);
      // Instantiate an empty background model
      Mat bg_model(frame.rows, frame.cols, CV_8UC3);

      // Apply BGS algorithm on the current frame
      subtractor->process(frame, seg_map, bg_model);
    }
  }

  // Delete the instantiated BGS algorithm
  delete subtractor;
}
```

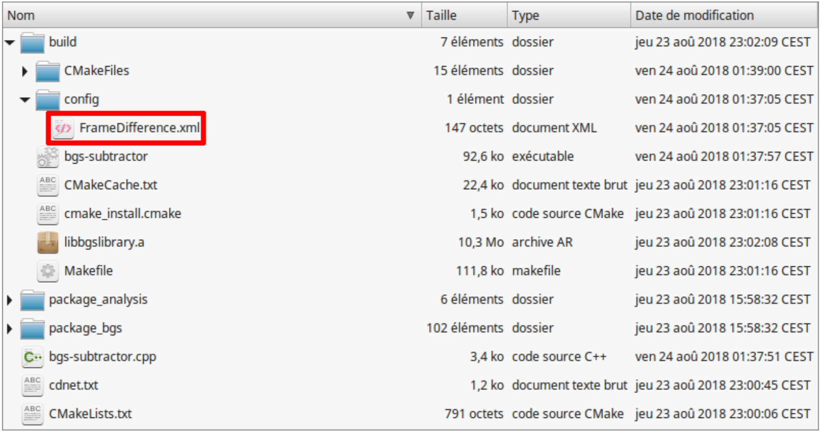
Compilation and Execution



```
~/Bureau/cdnet-bgs/build $ ./bgs-subtractor
badWeather/blizzard
900 - 7000
FrameDifference()
```

- You can compile (resp. launch) the code as in the slide 20 (resp. 27).
- If everything is correct, a window displaying the segmentation maps appears.

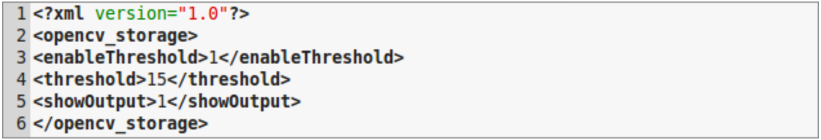
A New File Appeared!



Nom	Taille	Type	Date de modification
build	7 éléments	dossier	jeu 23 aoû 2018 23:02:09 CEST
CMakeFiles	15 éléments	dossier	ven 24 aoû 2018 01:39:00 CEST
config	1 élément	dossier	ven 24 aoû 2018 01:37:05 CEST
FrameDifference.xml	147 octets	document XML	ven 24 aoû 2018 01:37:05 CEST
bgs-subtractor	92,6 ko	exécutable	ven 24 aoû 2018 01:37:57 CEST
CMakeCache.txt	22,4 ko	document texte brut	jeu 23 aoû 2018 23:01:16 CEST
cmake_install.cmake	1,5 ko	code source CMake	jeu 23 aoû 2018 23:01:16 CEST
libbgslibrary.a	10,3 Mo	archive AR	jeu 23 aoû 2018 23:02:08 CEST
Makefile	111,8 ko	makefile	jeu 23 aoû 2018 23:01:16 CEST
package_analysis	6 éléments	dossier	jeu 23 aoû 2018 15:58:32 CEST
package_bgs	102 éléments	dossier	jeu 23 aoû 2018 15:58:32 CEST
bgs-subtractor.cpp	3,4 ko	code source C++	ven 24 aoû 2018 01:37:51 CEST
cdnet.txt	1,2 ko	document texte brut	jeu 23 aoû 2018 23:00:45 CEST
CMakeLists.txt	791 octets	code source CMake	jeu 23 aoû 2018 23:00:06 CEST

- We used the frame difference algorithm in our code.
- A file `FrameDifference.xml` appeared in the `config` folder.

Configuration of a BGS Algorithm



```xml
<?xml version="1.0"?>
<opencv_storage>
<enableThreshold>1</enableThreshold>
<threshold>15</threshold>
<showOutput>1</showOutput>
</opencv_storage>
```

- An XML file is automatically created the first time a BGS algorithm is used.
- This file enables to tune the parameters of the frame difference.
- This tuning must be done before launching our `bgs-subtractor`.
- For instance, the threshold can be modified by tuning the value surrounded by the `<threshold>` tag (here, the value is `15`).

Using Another BGS Algorithm is Easy!

- To use another BGS algorithm, we must change a unique line:

```
IBGS* subtractor = new FrameDifference;
```

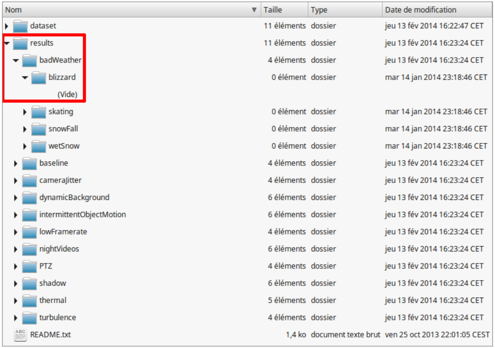
- For instance, to use $\Sigma - \Delta$, we can modify this line as follows:

```
IBGS* subtractor = new SigmaDelta;
```

- You can find the available algorithms in the file `package_bgs/bgslibrary.h`.

Step 6: Saving the Segmentation Maps

Procedure



Nom	Taille	Type	Date de modification
dataset	11 éléments	dossier	jeu 13 fév 2014 16:22:47 CET
results	11 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
badWeather	4 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
blizzard	0 élément	dossier	mar 14 jan 2014 23:18:46 CET
(Vide)			
skating	0 élément	dossier	mar 14 jan 2014 23:18:46 CET
snowFall	0 élément	dossier	mar 14 jan 2014 23:18:46 CET
wetSnow	0 élément	dossier	mar 14 jan 2014 23:18:46 CET
baseline	4 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
cameraJitter	4 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
dynamicBackground	6 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
intermittentObjectMotion	6 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
lowFramerate	4 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
nightVideos	6 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
PTZ	4 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
shadow	6 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
thermal	5 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
turbulence	4 éléments	dossier	jeu 13 fév 2014 16:23:24 CET
README.txt	1,4 ko	document texte brut	ven 25 oct 2013 22:01:05 CEST

- For assessing a BGS algorithm, we must save the results.
- Specifically, we must save the maps produced during a temporal ROI.
- They can be saved in the `results` folder of the CDnet dataset.
- It contains empty category and sequence folders.
- The name of a map is `bin`, the frame number encoded with 6 digits, and `.png`.

Modification of `bgs-subtractor.cpp`

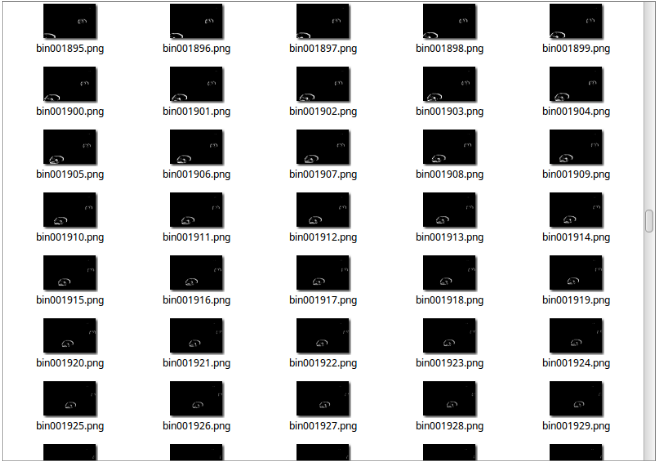
Add the path to the CDnet results.

```
...
#define RESULTS_PATH "/home/user/Bureau/dataset2014/results"
```

Code to Save the Segmentation Maps in `bgs-subtractor.cpp`

```cpp
...
int main() {
  ...
  // For each sequence
  for (size_t seq_idx = 0; seq_idx < seqs.size(); ++seq_idx) {
    ...
    // For each frame
    for (int f_num = 1; f_num <= frame_end; ++f_num) {
      ...
      // If we are in the evaluation period
      if (f_num >= frame_begin) {
        // Stream to format the output image file name
        stringstream write_path;
        /* Path to the output image file containing the
         * current segmentation map
         */
        write_path << RESULTS_PATH << "/" << seqs[seq_idx]
                   << "/bin" << setw(6) << setfill('0')
                   << f_num << ".png";
        // Ask OpenCV to write the segementation map in the file
        imwrite(write_path.str(), seg_map);
      }
    }
  }
}
```

Compilation and Execution



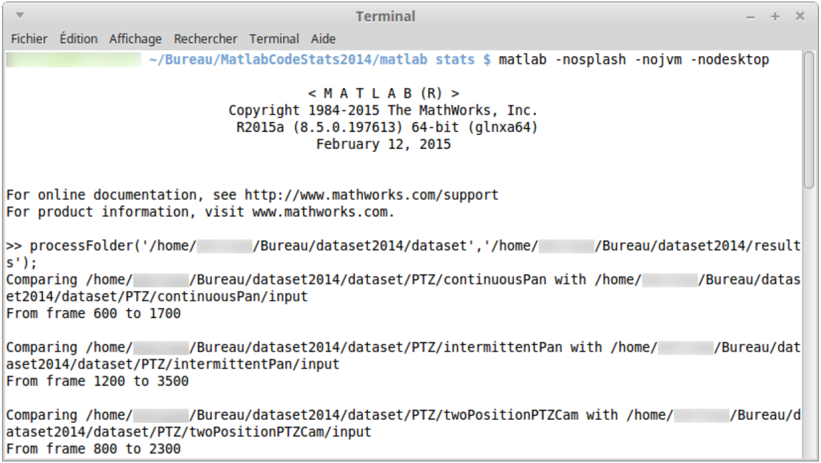
- You can compile (resp. launch) the code as in the slide 20 (resp. 27).
- If everything is correct, the maps should be saved in the `results` folder.

Performance Evaluation

What’s Next?

- We want to assess a given BGS algorithm on the CDnet dataset.
- We compare the resulting segmentation maps with the ground-truth.
- The result of such a comparison is expressed by metrics/scores.
- An evaluation tool computing those metrics is given with CDnet (see slide 11).
- We will see how to use the Matlab version of this tool.
- Feel free to use the Python version if it is more convenient to you!

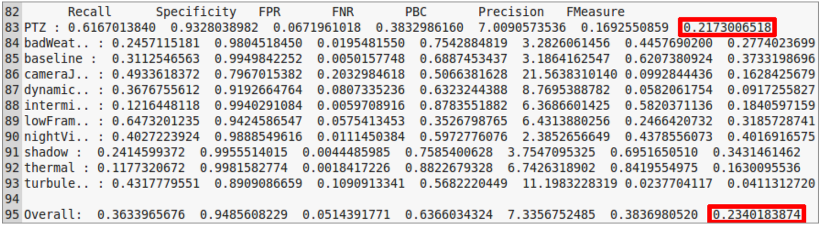
Procedure



```
[redacted] ~/Bureau/MatlabCodeStats2014/matlab stats $ matlab -nosplash -nojvm -nodesktop

                              < M A T L A B (R) >
                    Copyright 1984-2015 The MathWorks, Inc.
                     R2015a (8.5.0.197613) 64-bit (glnxa64)
                               February 12, 2015


For online documentation, see http://www.mathworks.com/support
For product information, visit www.mathworks.com.

>> processFolder('/home/[redacted]/Bureau/dataset2014/dataset','/home/[redacted]/Bureau/dataset2014/result
s');
Comparing /home/[redacted]/Bureau/dataset2014/dataset/PTZ/continuousPan with /home/[redacted]/Bureau/datas
et2014/dataset/PTZ/continuousPan/input
From frame 600 to 1700

Comparing /home/[redacted]/Bureau/dataset2014/dataset/PTZ/intermittentPan with /home/[redacted]/Bureau/dat
aset2014/dataset/PTZ/intermittentPan/input
From frame 1200 to 3500

Comparing /home/[redacted]/Bureau/dataset2014/dataset/PTZ/twoPositionPTZCam with /home/[redacted]/Bureau/d
ataset2014/dataset/PTZ/twoPositionPTZCam/input
From frame 800 to 2300
```

- In Matlab, use the function `processFolder()`.
- The first parameter is the path to the CDnet dataset.
- The second parameter is the path to the results to assess.

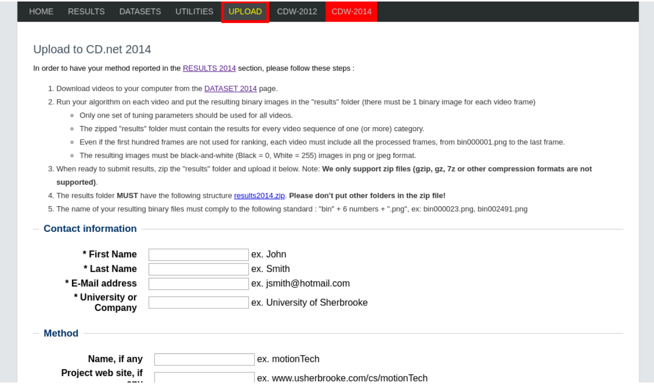
Metrics



```
      Recall       Specificity   FPR         FNR         PBC         Precision   FMeasure
PTZ : 0.6167013840  0.9328038982  0.0671961018  0.3832986160  7.0090573536  0.1692550859  0.2173006518
badWeat.. : 0.2457115181  0.9804518450  0.0195481550  0.7542884819  3.2826061456  0.4457690200  0.2774023699
baseline :  0.3112546563  0.9949842252  0.0050157748  0.6887453437  3.1864162547  0.6207380924  0.3733198696
cameraJ.. : 0.4933618372  0.7967015382  0.2032984618  0.5066381628  21.5638310140 0.0992844436  0.1628425679
dynamic.. : 0.3676755612  0.9192664764  0.0807335236  0.6323244388  8.7695388782  0.0582061754  0.0917255827
intermi.. : 0.1216448118  0.9940291084  0.0059708916  0.8783551882  6.3686601425  0.5820371136  0.1840597159
lowFram.. : 0.6473201235  0.9424586547  0.0575413453  0.3526798765  6.4313880256  0.2466420732  0.3185728741
nightVi.. : 0.4027223924  0.9888549616  0.0111450384  0.5972776076  2.3852656649  0.4378556073  0.4016916575
shadow :  0.2414599372  0.9955514015  0.0044485985  0.7585400628  3.7547095325  0.6951650510  0.3431461462
thermal : 0.1177320672  0.9981582774  0.0018417226  0.8822679328  6.7426318902  0.8419554975  0.1630095536
turbule.. : 0.4317779551  0.8909086659  0.1090913341  0.5682220449  11.1983228319 0.0237704117  0.0411312720

Overall:  0.3633965676  0.9485608229  0.0514391771  0.6366034324  7.3356752485  0.3836980520  0.2340183874
```

- When `processFolder()` is over, a `cm.txt` file is generated.
- The file is located in the dataset folder.
- For each category, it contains the scores averaged over sequences.
- It contains also the scores averaged over all CDnet sequences.
- For instance, for the F. Diff., F1 is $\simeq 0.22$ on PTZ, and $\simeq 0.23$ on the dataset.

Comprehensive Evaluation



Upload to CD.net 2014

In order to have your method reported in the RESULTS 2014 section, please follow these steps :

1. Download videos to your computer from the DATASET 2014 page.
2. Run your algorithm on each video and put the resulting binary images in the "results" folder (there must be 1 binary image for each video frame)
   - Only one set of tuning parameters should be used for all videos.
   - The zipped "results" folder must contain the results for every video sequence of one (or more) category.
   - Even if the first hundred frames are not used for ranking, each video must include all the processed frames, from bin000001.png to the last frame.
   - The resulting images must be black-and-white (Black = 0, White = 255) images in png or jpeg format.
3. When ready to submit results, zip the "results" folder and upload it below. Note: **We only support zip files (gzip, gz, 7z or other compression formats are not supported)**.
4. The results folder **MUST** have the following structure results2014.zip. **Please don't put other folders in the zip file!**
5. The name of your resulting binary files must comply to the following standard : "bin" + 6 numbers + ".png", ex: bin000023.png, bin002491.png

Contact information

*** First Name** ex. John

*** Last Name** ex. Smith

*** E-Mail address** ex. jsmith@hotmail.com

*** University or Company** ex. University of Sherbrooke

Method

Name, if any ex. motionTech

Project web site, if ex. www.usherbrooke.com/cs/motionTech

- If your results are convincing, you can upload them on the CDnet website.
- This enables to discover your position in the ranking.
- Moreover, the website performs a deeper evaluation.
- More ground-truth is available internally on the server (for avoiding cheating).
- Your results are even more convincing? Let's publish your paper!

Integrate Your Own Algorithm

Procedure

- Until now, we saw how to apply a BGS algorithm from the BGSLibrary on CDnet.
- Also, we saw how to assess quantitatively the results.
- New question: Is it possible to do those operations with your own algorithm?
- Answer: Yes!
- The solution is to integrate your algorithm to the BGSLibrary.
- Consists in creating a class (e.g. `MyAlgo`) inheriting from the class `IBGS`.
- `MyAlgo` must be in the namespace `bgslibrary::algorithms`.
- In `MyAlgo`, you must override the relevant methods of `IBGS`.
- You can start this work by creating a header and implementation file (`MyAlgo.h` and `MyAlgo.cpp`) in `package_bgs`.

Methods to Override

- In your class inheriting from `IBGS`, you must override 3 methods:

```
void process(const cv::Mat& img_input,
             cv::Mat& img_output,
             cv::Mat& img_bgmodel);
void saveConfig();
void loadConfig();
```

- `process()` applies the algorithm on the current frame. The parameters are:

`img_input`	The current frame (input).
`img_output`	The resulting segmentation map (output).
`img_bgmodel`	An image representing the current background model (output).

- `saveConfig()` saves the parameter values in the algorithm XML config file.

- `loadConfig()` loads the parameter values in the algorithm XML config file.

- Let's analyze the code of the frame difference!

Header FrameDifference.hpp

```
#pragma once
#include "IBGS.h"

namespace bgslibrary {
  namespace algorithms {
    // Inherits from IBGS
    class FrameDifference : public IBGS {
      private:
        bool enableThreshold;                  // Parameter 1
        int threshold;                         // Parameter 2

      public:
        FrameDifference();                     // Constructor
        ~FrameDifference();                    // Destructor
        void process(const cv::Mat &img_input, // Method 1
                     cv::Mat &img_output,
                     cv::Mat &img_bgmodel);

      private:
        void saveConfig();                     // Method 2
        void loadConfig();                     // Method 3
    };
  }
}
```

Constructor and Destructor `FrameDifference.hpp`

```
#include "FrameDifference.h"

using namespace bgslibrary::algorithms;

// Constructor
FrameDifference::FrameDifference() :
// Parameters default values
enableThreshold(true), threshold(15) {
  // Display the name of the instantiated algorithm
  std::cout << "FrameDifference()" << std::endl;
  /* Call the setup function of the BGSLibrary to initialize
   * the algorithm XML config file
   */
  setup("./config/FrameDifference.xml");
}

// Destructor
FrameDifference::~FrameDifference() {
  // Display the name of the destroyed algorithm
  std::cout << "~FrameDifference()" << std::endl;
  // Nothing to destroy for this algorithm
}
```

Implementation of the BGS Algorithm in `FrameDifference.hpp`

```
void FrameDifference::process(const cv::Mat &img_input,
                              cv::Mat &img_output,
                              cv::Mat &img_bgmodel) {
  /* Call the initialization function of the BGSLibrary to
   * allocate memory related to img_output and img_bgmodel
   * whether they are empty.
   */
  init(img_input, img_output, img_bgmodel);

  // If internal background model is empty (first frame)
  if (img_background.empty()) {
    // Copy the first frame as background model
    img_input.copyTo(img_background);
    // Stop (we cannot detect motion from a unique frame)
    return;
  }

  // Absolute difference between model and current frame
  cv::absdiff(img_background, img_input, img_foreground);

  // If input frame is RGB
  if (img_foreground.channels() == 3)
    // Convert it to grayscale
    cv::cvtColor(img_foreground, img_foreground, CV_BGR2GRAY);
    ...
```

Implementation of the BGS Algorithm in FrameDifference.hpp (2)

```
  ...
  // If threshold operation is required (yes by default)
  if (enableThreshold)
    /* Apply threshold on input frame and save it as the
     * internal segmentation map
     */
    cv::threshold(img_foreground, img_foreground, threshold,
                  255, cv::THRESH_BINARY);

// Code to show the segmentation map in a graphical window
#ifndef MEX_COMPILE_FLAG
  if (showOutput)
    // Give the name of your algorithm to the graphical window
    cv::imshow("Frame Difference", img_foreground);
#endif

  // Copy the internal segmentation map to the output one
  img_foreground.copyTo(img_output);
  // Copy the input frame as the internal background model
  img_input.copyTo(img_background);
  // Copy the internal background model as the output one
  img_background.copyTo(img_bgmodel);
  // The first frame has been processed
  firstTime = false;
}
```

Code to Save XML Config File in `FrameDifference.hpp`

```
void FrameDifference::saveConfig() {
  // Ask OpenCV to open the XML file to write
  CvFileStorage* fs = cvOpenFileStorage(config_xml.c_str(),
                                        nullptr,
                                        CV_STORAGE_WRITE);

  // Write enableThreshold parameter value as an integer
  cvWriteInt(fs, "enableThreshold", enableThreshold);
  // Write threshold parameter value as an integer
  cvWriteInt(fs, "threshold", threshold);
  // Write showOutput parameter value as an integer
  cvWriteInt(fs, "showOutput", showOutput);

  // Writing is over (closing)
  cvReleaseFileStorage(&fs);
}
```

Note that OpenCV limits the parameter types that can be written. You can use:

- `cvWriteInt` to write an integer parameter.
- `cvWriteReal` to write a floating-point parameter.
- `cvWriteString` to write a string parameter.

Code to Load XML Config File in `FrameDifference.hpp`

```cpp
void FrameDifference::loadConfig() {
  // Ask OpenCV to open the XML file to read
  CvFileStorage* fs = cvOpenFileStorage(config_xml.c_str(),
                                        nullptr,
                                        CV_STORAGE_READ);

  // Read enableThreshold as an integer (true if not defined)
  enableThreshold = cvReadIntByName(fs, nullptr,
                                    "enableThreshold", true);
  // Read threshold as an integer (15 if not defined)
  threshold = cvReadIntByName(fs, nullptr, "threshold", 15);
  // Read showOutput as an integer (true if not defined)
  showOutput = cvReadIntByName(fs, nullptr, "showOutput", true);

  // Reading is over (closing)
  cvReleaseFileStorage(&fs);
}
```

Note that, once again OpenCV limits the parameter types that can be read to integers, floating-points, and strings.

The Journey is Coming to an End

- Do not forget to add your algorithm into `bgslibrary.h`.

- You can send it to Andrews Sobral to be integrated in the official BGSLibrary.

- You are now able to:
  - Integrate your own BGS algorithm in the BGSLibray.
  - Apply it on the CDnet dataset.
  - Assess it with metrics/scores.

شكراً جزيلاً
thank you
danke
謝謝
рахмат
ngiyabonga
teşekkür ederim
gracias
merci
obrigado
спасибо
Баярлалаа
bedankt
dziękuję
go raibh maith agat
mochchakkeram
sukriya
kop khun krap
grazie
arigatô
takk
dakujem
terima kasih
감사합니다
ありがとう
ευχαριστώ
tapadh leat
dank je
mersi
barka
welalin
tack
spas
kia ora
faafetai lava
vinaka
спасибі
blagodaram
misaotra
matondo
paldies
grazzi
mahalo
хвала
asante
manana
obrigada
tenki
murakoze
chokrane
mamnun
дякую
djiere dieuf
tau
kiitos
dankie
nanni
nandri
dhanyavad
hvala
maururu
köszönöm
bayarlalaa
gràcie
enkosi
sobodi
dėkuji
mèsi
didi madloba
sagolun
najis tuke
kam sah hamnida
তোমাকে ধন্যবাদ
rahmat
chnorakaloutioun
gratias ago
gràcies
sulpáy
akun dankon
aciū
taiku
tanemirt
rahmet
diolch
dhanyavadagalu
shukriya
mercê
мерси
trugarez
xièxie

References

References

[1] T. Bouwmans et al., eds. **Background Modeling and Foreground Detection for Video Surveillance**. Chapman and Hall/CRC, 2014. ISBN: 9781482205374.

[2] N. Goyette et al. "changedetection.net: A New Change Detection Benchmark Dataset." In: **IEEE International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)**. 2012. DOI: `10.1109/CVPRW.2012.6238919`.

[3] A. Sobral. "BGSLibrary: An OpenCV C++ Background Subtraction Library." In: **Workshop de Visao Computacional (WVC)**. Rio de Janeiro, Brazil, 2013, pp. 1–6. DOI: `10.13140/2.1.1740.7044`.

[4] Y. Wang et al. "CDnet 2014: An Expanded Change Detection Benchmark Dataset." In: **IEEE International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)**. Columbus, Ohio, USA, 2014, pp. 393–400. DOI: `10.1109/CVPRW.2014.126`.