

CVVisual

20. Juni 2014

Inhaltsverzeichnis

0.1	Gliederung	2
1	Einführung in OpenCV	2
1.1	Überblick	2
1.2	Matrizen	3
1.3	Filter	3
1.4	Filter	3
1.5	Filter	4
1.6	Filter	4
1.7	Matches	4
1.8	Matches	4
2	Motivation	4
2.1	Debuggen von OpenCV	4
2.2	Ziele	8
3	Anwenderfeatures	8
3.1	Verwendung	8
3.2	Übersicht	8
3.3	Übersicht	10
3.4	Übersicht	10
3.5	Übersicht	10
3.6	Übersicht	10
3.7	Filter	10
3.8	Filter	14
3.9	Matches	14
3.10	Matches	14
4	GUI-Demo	14

5 Architektur	14
5.1 Entwurf	14
5.2 Signals/Slots & Templates	16
5.3 RegisterHelper	16
5.4 (Auto-)FilterWidget	17
5.5 ZoomableImage	17
5.6 MatchScene	17
5.7 Match/KeyPointSetting	18
5.8 Views	18
6 Dokumentation	18
6.1 Tutorials, Beispiele	18
6.2 Kurzdokumentation	18
6.3 Referenz:	21
7 API	22
7.1 Anwender API	22
7.2 Interne API	22
8 Ausblick	22
8.1 Rezeption	22
8.2 Rezeption	22
8.3 Links	22

0.1 Gliederung

- Einführung in OpenCV
- Motivation
- Anwenderfeatures
- Gui-Demo
- Dokumentation
- Architektur
- API
- Ausblick

1 Einführung in OpenCV

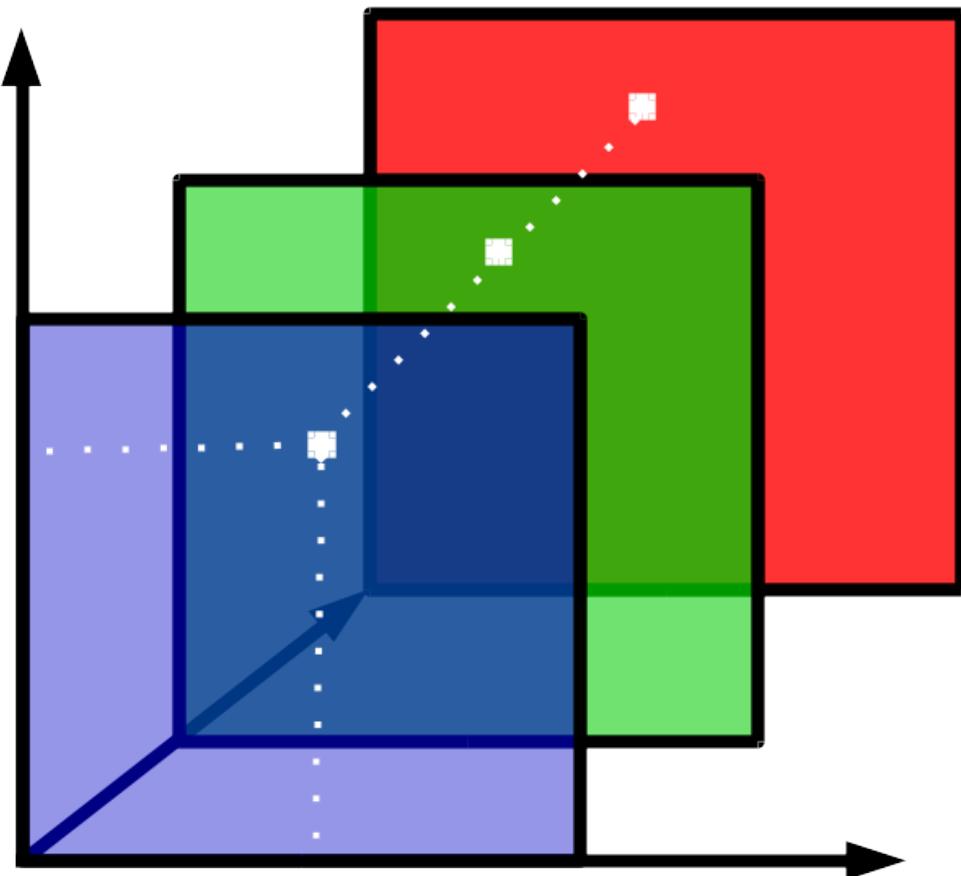
1.1 Überblick

- Bildverarbeitung
- weite Verbreitung
- Matrizen als Grundlage

- Filter + Matches

1.2 Matrizen

Bild = mehrdimensionale Matrix



Bsp. BGR-Bild: 1. Channel blau, 2. Channel grün usw.

1.3 Filter

Berechnung auf Umgebung jedes Pixels

1.4 Filter

Beispiel dilate: helle Flächen werden größer

5	7	3	5	5	5
3	2	6	7	6	5
2	3	2	4	6	6
3	3	5	6	4	5
1	4	6	2	2	4
3	4	7	5	6	5



1.5 Filter

Beispiel dilate: helle Flächen werden größer

1.6 Filter

Beispiel dilate: helle Flächen werden größer

1.7 Matches

Keypoints = charakteristische Punkte

1.8 Matches

Match = Paar aus Keypoints

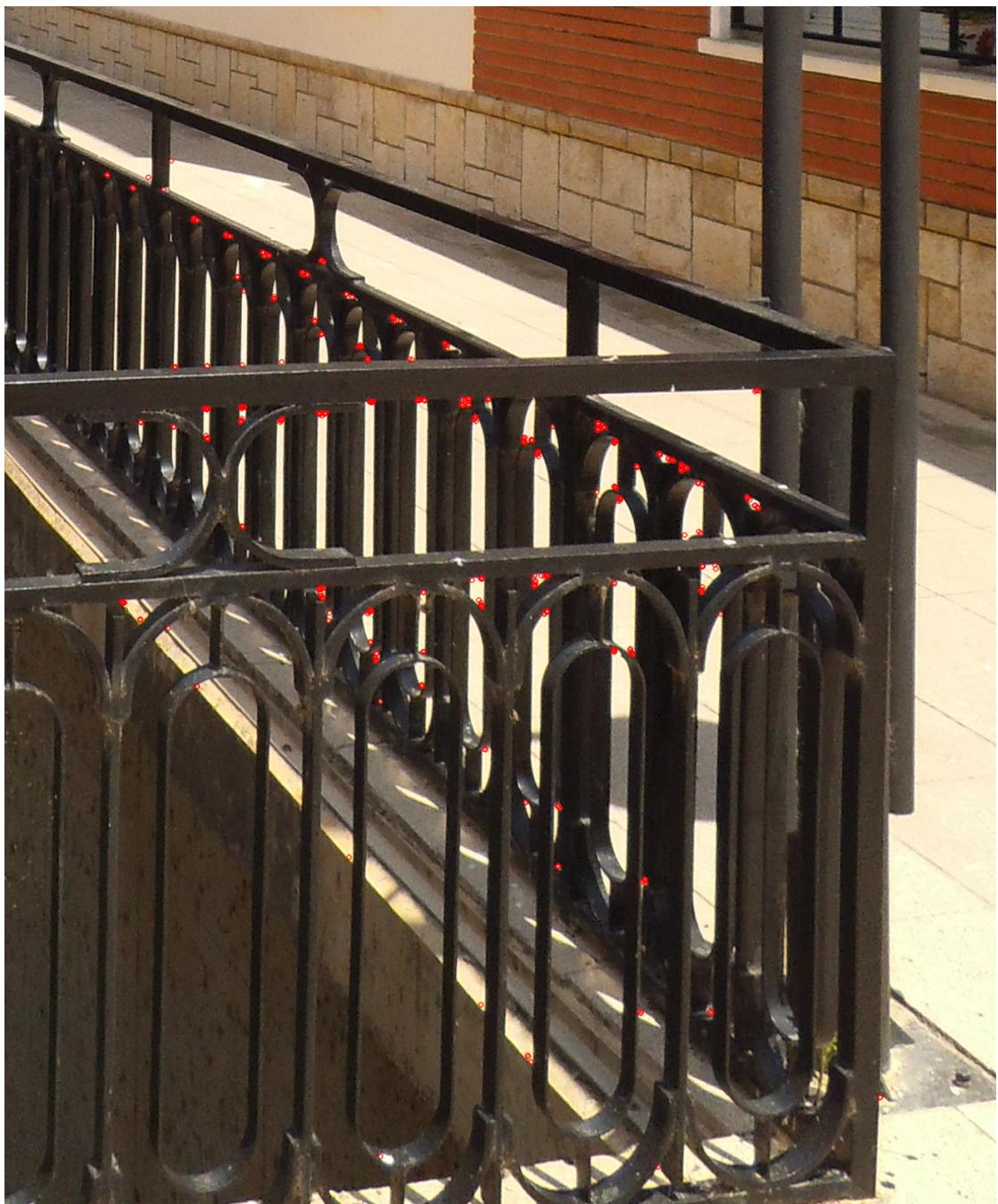
2 Motivation

2.1 Debuggen von OpenCV

Systematisches Debugging statt „Random Code“

Hinweis auf showMatches/showKeypoints







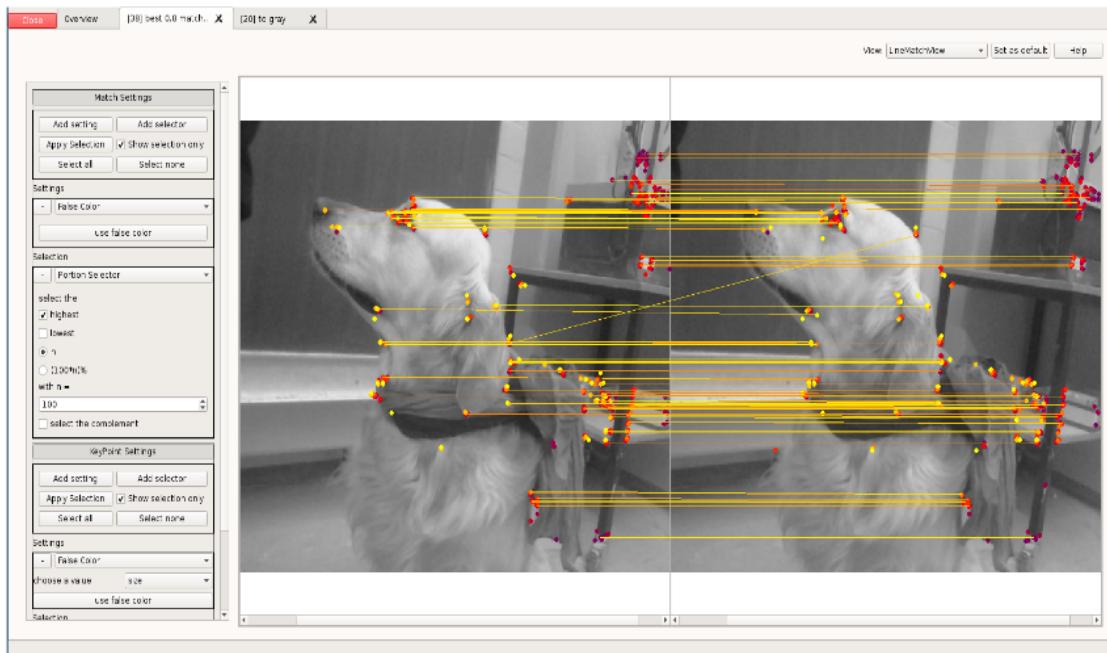
```
#ifdef DEBUG
    Mat img_matches;
    drawMatches( img_1, keypoints_1, img_2, keypoints_2,
                good_matches, img_matches, Scalar::all(-1), Scalar::all(-1),
                vector<char>(), DrawMatchesFlags::NOT_DRAW_SINGLE_POINTS );
    imshow("good matches", img_matches);
#endif
```

versus

```
cvv::debugMatches(img1, img2, keypoints_1, keypoints_2, good_matches);
```

2.2 Ziele

Visualisierung von Matritzen, Filtereffekten und Matches



3 Anwenderfeatures

3.1 Verwendung

```
std::string imgIdString{"imgRead"};
imgIDString += toString(imgId);
cvv::showImage(imgRead, CVVISUAL_LOCATION, imgIdString);

// convert to grayscale:
cv::Mat imgGray;
cv::cvtColor(imgRead, imgGray, CV_BGR2GRAY);
cvv::debugFilter(imgRead, imgGray, CVVISUAL_LOCATION,
    "to gray", "SingleFilterView");
```

3.2 Übersicht

Übersicht über alle Aufrufe

CVVisual | main window

Close **Overview**

No grouping specified, use #group to specify one

ID	Image 1	Image 2	Description	Function	File	Line	Type
1			IMG_1353.JPG	int main(int, char**)	/home/partimenerd/Studium/PSE/cvisual_test/main.cpp	22	singleImage
2			IMG_1130.JPG	int main(int, char**)	/home/partimenerd/Studium/PSE/cvisual_test/main.cpp	22	singleImage
3			IMG_1396.JPG	int main(int, char**)	/home/partimenerd/Studium/PSE/cvisual_test/main.cpp	22	singleImage
4			IMG_1397.JPG	int main(int, char**)	/home/partimenerd/Studium/PSE/cvisual_test/main.cpp	22	singleImage

Zoom ...

3.3 Übersicht

Filterbar

No grouping specified, use #group to specify one							
ID	Image 1	Image 2	Description	Function	File	Line	Type
19			<no description>	int main(int, char**)	/home/parttimenerd/Studium/PSE/cvisual_test/main.cpp	59	match
20			<no description>	int main(int, char**)	/home/parttimenerd/Studium/PSE/cvisual_test/main.cpp	59	match
21			<no description>	int main(int, char**)	/home/parttimenerd/Studium/PSE/cvisual_test/main.cpp	59	match
22			<no description>	int main(int, char**)	/home/parttimenerd/Studium/PSE/cvisual_test/main.cpp	59	match

3.4 Übersicht

Sortierbar

3.5 Übersicht

Gruppierbar

3.6 Übersicht

3.7 Filter

- 2 Bilder → 1 Bild
- Differenzbilder, Overlay, geänderte Pixel für Filter

CVVisual | main window

[Close](#) [Overview](#)

#sort by line desc

Help

No grouping specified, use #group to specify one

ID	Image 1	Image 2	Description	Function	File	Line	Type
19			<no description>	int main(int, char**)	/home/partimenerd/Studium/PSE/cvisual_test/main.cpp	59	match
20			<no description>	int main(int, char**)	/home/partimenerd/Studium/PSE/cvisual_test/main.cpp	59	match
21			<no description>	int main(int, char**)	/home/partimenerd/Studium/PSE/cvisual_test/main.cpp	59	match
22			<no description>	int main(int, char**)	/home/partimenerd/Studium/PSE/cvisual_test/main.cpp	59	match

Zoom 100%

CVVisual | main window

#group by description

ID	Image 1	Description	Function	File	Line	Type	
5		IMG_1454.JPG	int main(int, char**)	/home/partimenerd/Studium/PSE/cvvisual_test/main.cpp	22	singleImage	
6		IMG_1455.JPG	int main(int, char**)	/home/partimenerd/Studium/PSE/cvvisual_test/main.cpp	22	singleImage	
erode							
7			erode	int main(int, char**)	/home/partimenerd/Studium/PSE/cvvisual_test/main.cpp	36	filter

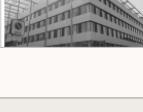
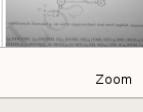
Zoom

CVVisual | main window

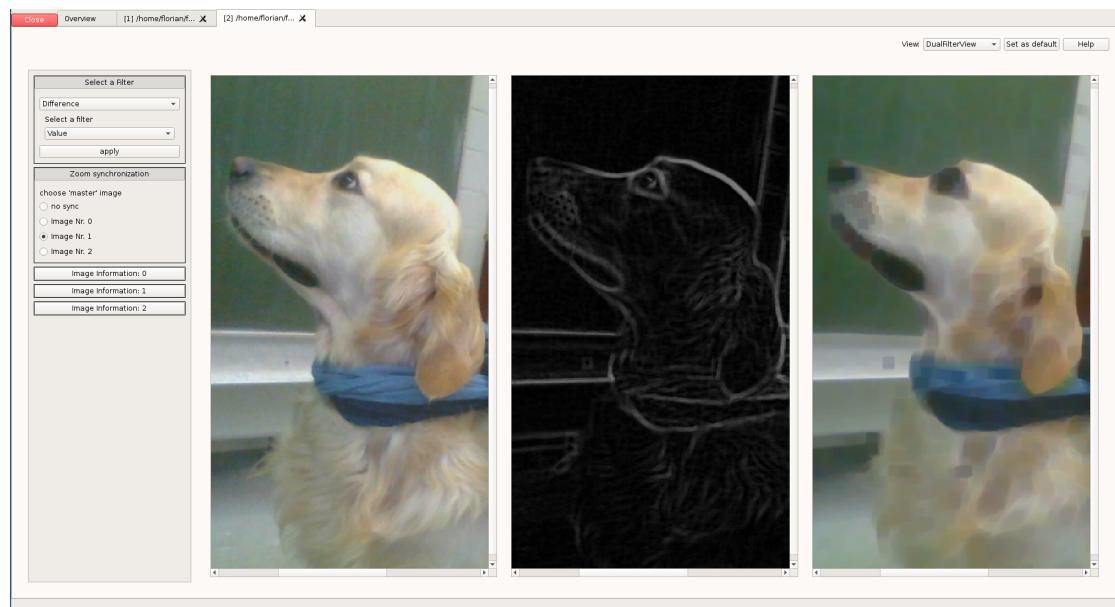
Close **Overview**

#group by description #sort by line desc #type match **Help**

<no description>

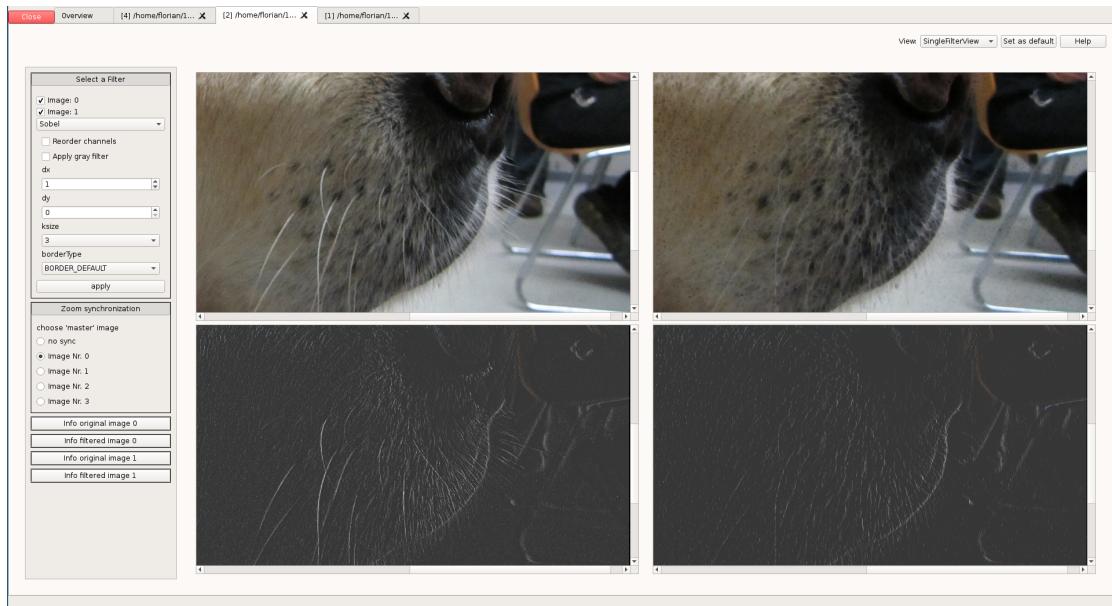
ID	Image 1	Image 2	Description	Function	File	Line	Type
19			<no description>	int main(int, char**)	/home/parttimenerd/Studium/PSE/cvisual_test/main.cpp	59	match
20			<no description>	int main(int, char**)	/home/parttimenerd/Studium/PSE/cvisual_test/main.cpp	59	match
21			<no description>	int main(int, char**)	/home/parttimenerd/Studium/PSE/cvisual_test/main.cpp	59	match
22			<no description>	int main(int, char**)	/home/parttimenerd/Studium/PSE/cvisual_test/main.cpp	59	match

Zoom 



3.8 Filter

- 1 Bild → 1 Bild
- Nachträgliche Anwendung weiterer Filter



3.9 Matches

- Anzeigen / Filtern von Keypoints / Matches
- Anzeige der Verbindungen von Keypoints

3.10 Matches

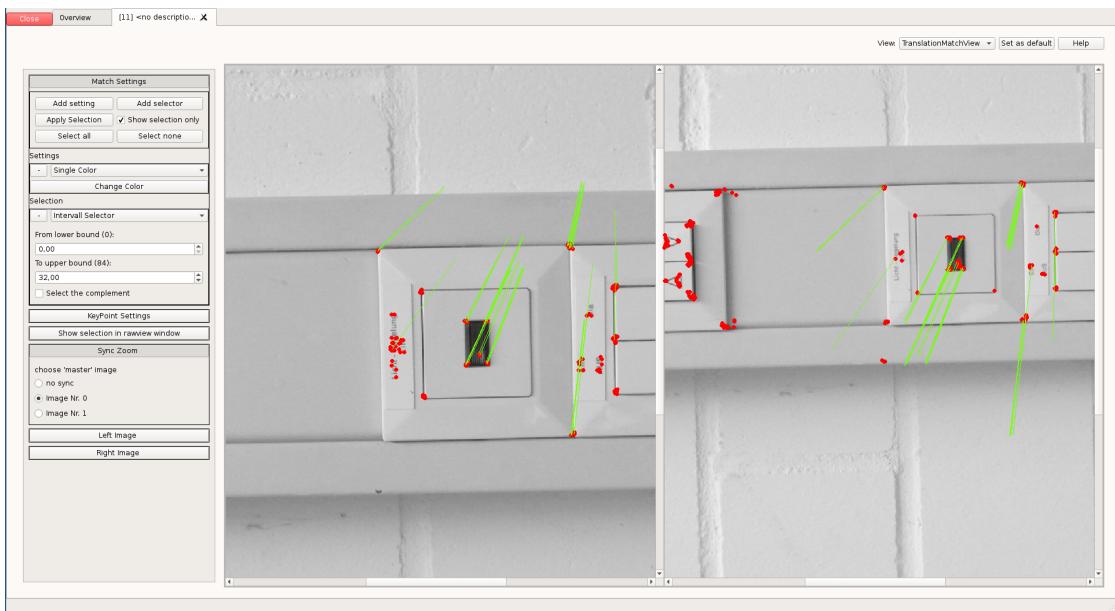
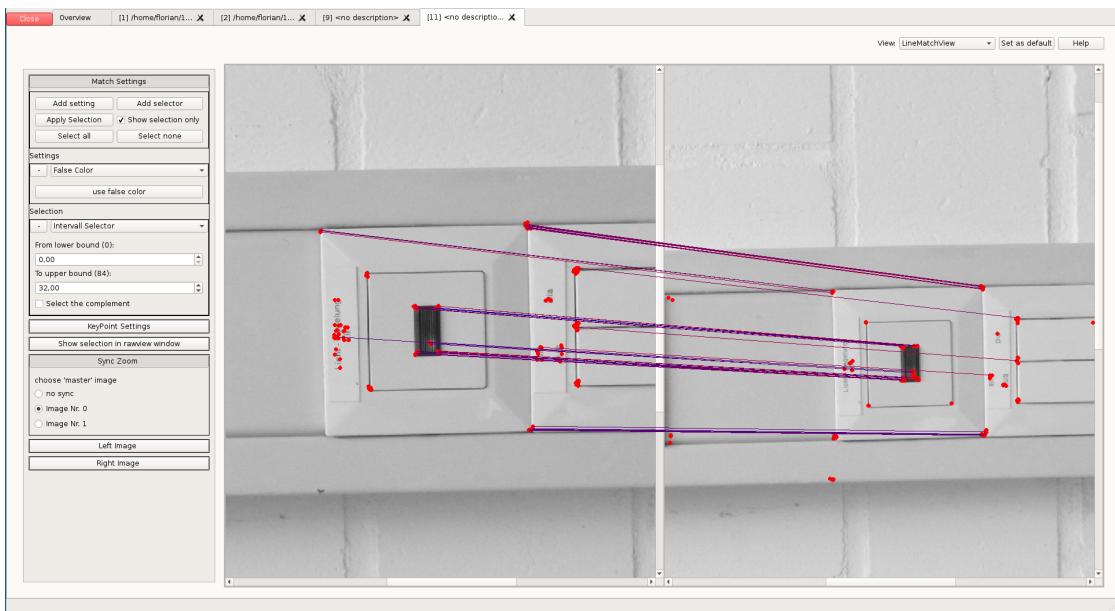
- Anzeigen / Filtern von Keypoints / Matches
- Anzeige der Translation von Keypoints

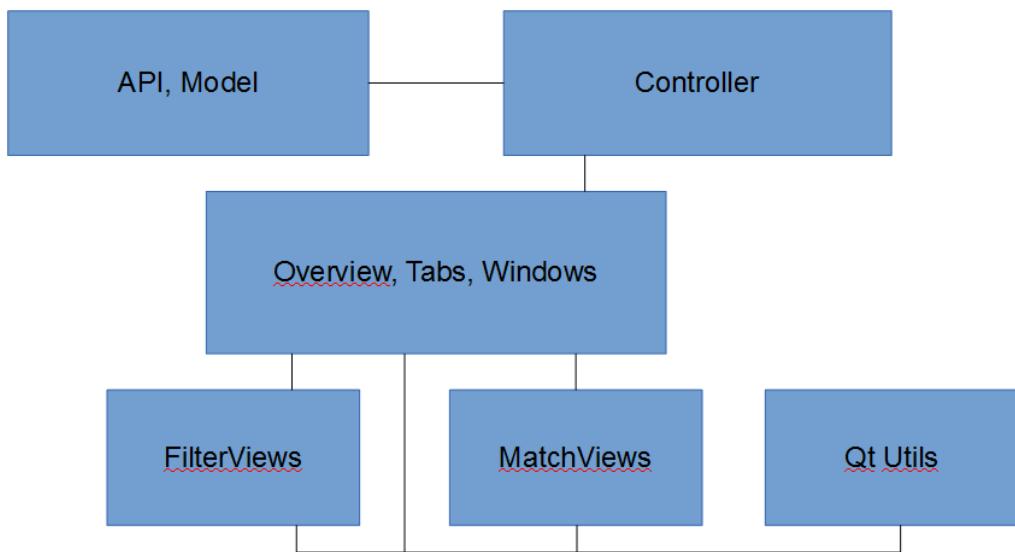
4 GUI-Demo

5 Architektur

5.1 Entwurf

- Trennung in API, Datenhaltung, Visualisierung





5.2 Signals/Slots & Templates

- Qt erlaubt keine Templateklassen mit Q_OBJECT
- Signals/Slots in Objekte ausgelagert

```

class SlotQString::public QObject
{   →     Q_OBJECT
..... public:
    →     SlotQString(const std::function<void(QString)> &f,
    →     →     →     ...QObject *parent = nullptr)
    →     →     →     ...QObject{parent}, function_{f}
    →     {   →     if (!f)
    →     →     →     throw std::invalid_argument{"invalidate function"};
    →     }
public slots:
    →     void slot(QString t) const
    →     →     {function_(t);}
..... private:
    →     std::function<void(QString)> function_;
};

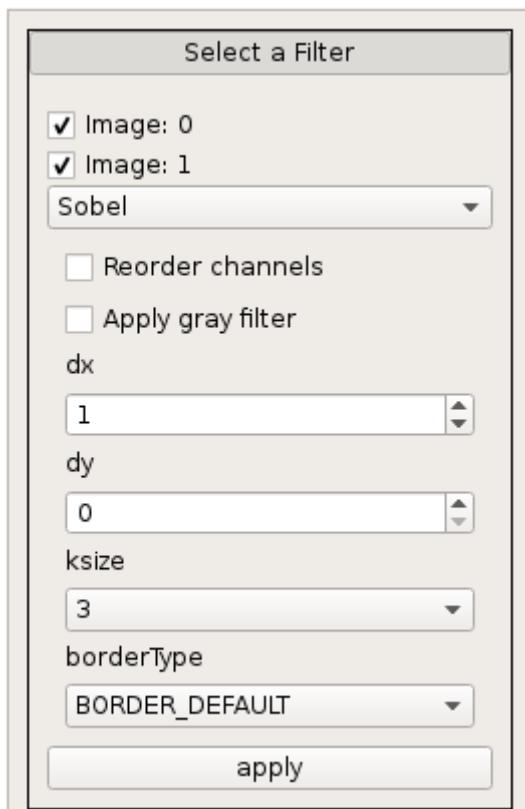
```

5.3 RegisterHelper

- Ermöglicht die Auswahl von Funktionen über eine Combobox
- Funktionen werden über eine API Funktion registriert

5.4 (Auto-)FilterWidget

- Unterklasse von RegisterHelper
- Ermöglicht Auswahl von Filtern
- Gibt Ergebnise per Signal weiter (z.B. an ein ZoomableImage)

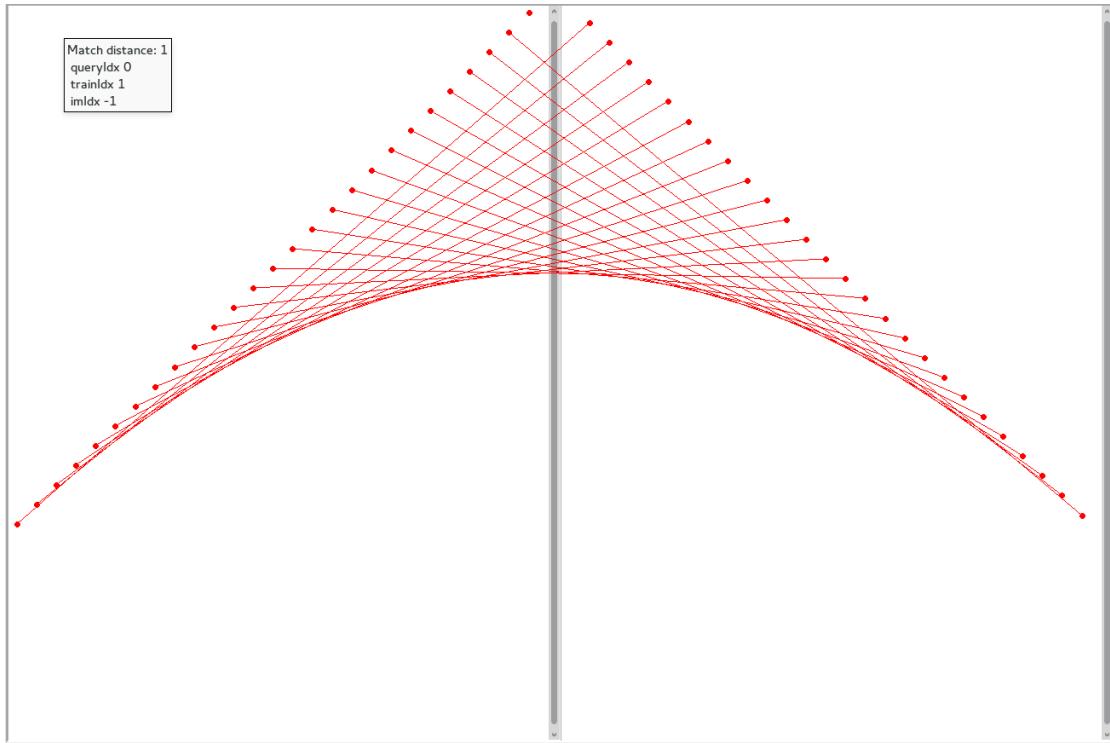


5.5 ZoomableImage

- Umwandlung von cv::Mat in Qt Format
- Signal & Slot für Zoom Events
- Slot zum Bild wechseln
- SyncZoomWidget erlaubt syncronen Zoom
- ZoomableImageOptionPanel zeigt weiter Informationen/Optinen an

5.6 MatchScene

- Enthält 2 ZoomableImages
- Enthält die KeyPoints/Matches als QGraphicsObjects



5.7 Match/KeyPointSetting

- Keine Auslagerung von Singals/Slots möglich
- Daher parallele Entwicklung von KeyPoint und MatchSetting
- Nur Selektierte KeyPoints/Matches werden angezeigt

5.8 Views

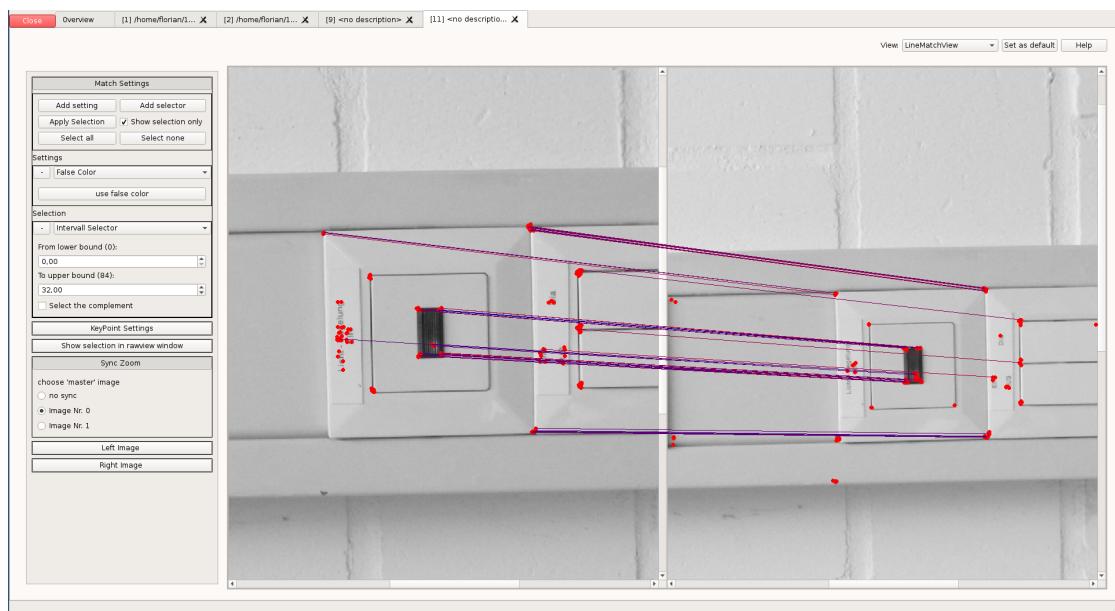
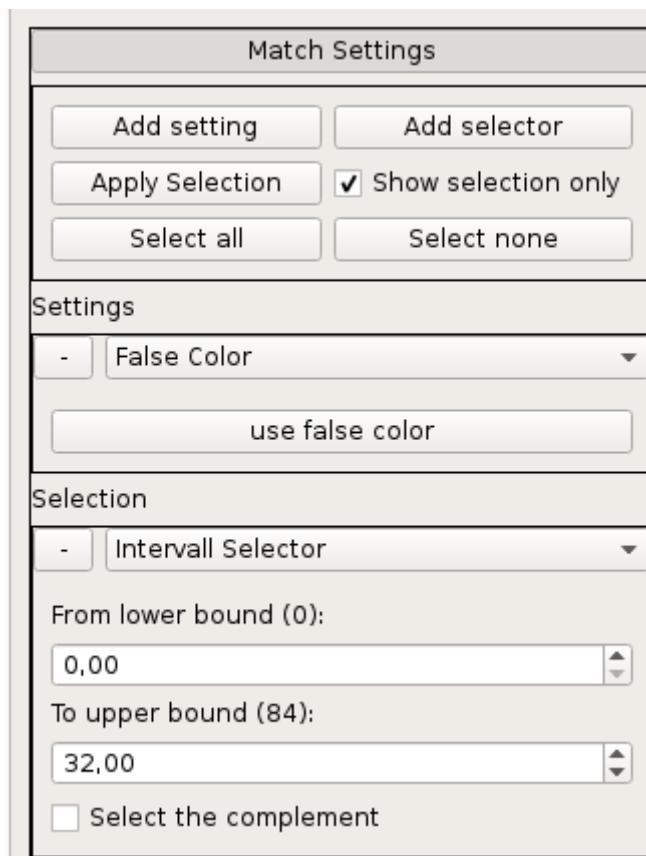
- Visualisierung der unterschiedlichen Aufrufe
- Unterscheiden sich meist in unterschiedlichen Nutzen von QT Util Klassen
- Einzige Aufgabe Weiterleitung und Annahme der Selektion (beim Wechsel der Views)

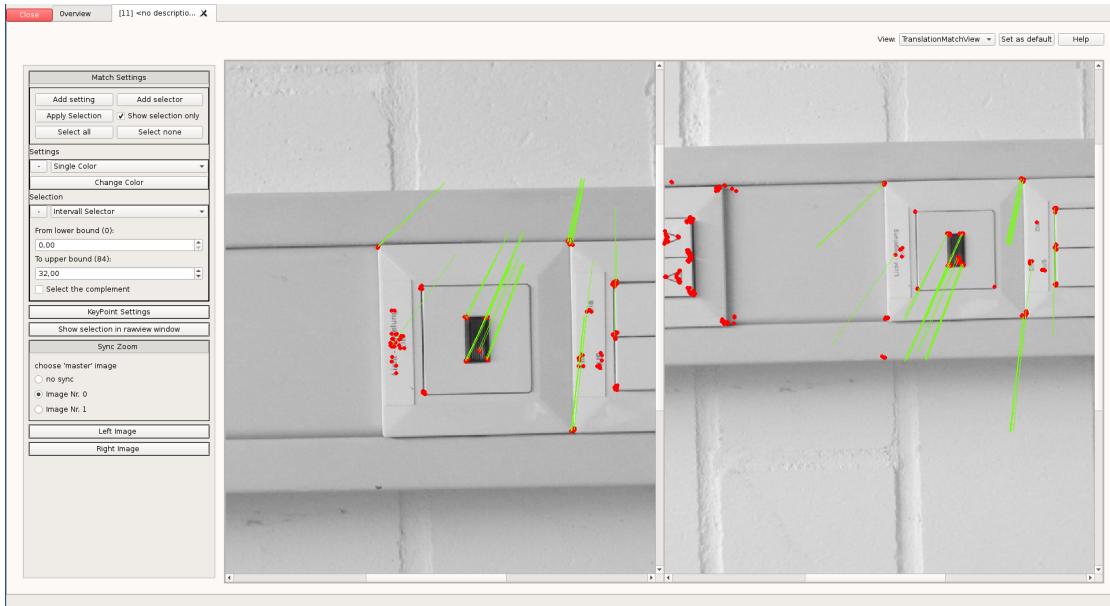
6 Dokumentation

6.1 Tutorials, Beispiele

6.2 Kurzdokumentation

Wird von der Hilfefunktion des Programms benutzt.





CVVisual : CVVisual Example

cv.mustlynerdless.de

CVVisual OpenCV debug visualization

[Home](#)

TUTORIAL

- [Introduction to using CVVisual](#)
- [Introduction to filter function widgets](#)
- [Über CVVisual](#)

REFERENCE

- [Views](#)
- [Filter query language](#)

[API Reference](#)

CVVisual Example

CVVisual is a debug visualization for OpenCV; thus, its main purpose is to offer different ways to visualize the results of OpenCV functions to make it possible to see whether they are what the programmer had in mind, and also to offer some functionality to try other operations on the images right in the debug window. This text wants to illustrate the use of CVVisual on a code example.

Image we want to debug this program:

[code_example/main.cpp](#)

Note the includes for CVVisual:

```

10 #include <opencv2/debug_mode.hpp>
11 #include <opencv2/show_image.hpp>
12 #include <opencv2/filter.hpp>
13 #include <opencv2/dmatch.hpp>
14 #include <opencv2/final_show.hpp>

```

It takes 10 snapshots with the webcam. With each, it first shows the image alone in the debug window,

```

97 cvv::showImage(imgRead, CVVISUAL_LOCATION, imgIdString.c_str());

```

then converts it to grayscale and calls CVVisual with the original and resulting image,

```

101 cv::cvtColor(imgRead, imgGray, CV_BGR2GRAY);

```

Home
 TUTORIAL
 Introduction to using CVVisual
 Introduction to filter function widgets
 Über CVVisual
 REFERENCE
 Views
 Filter query language
 API Reference

Views

General information:

Most views offer an `ImageInformation` collapsable in their accordion menus.
 The zoom can be found here.
`Ctrl + Mouse wheel` is also zoom; `Ctrl + Shift + Mouse wheel` is a slower zoom.
 If the zoom is deeper than 60%, the image's pixels will be overlaid with their channel values; usually, the order is `BGR[+alpha]` from the top.

Single Image View:

Associated with the `debugSingleImage()` function.
 Shows one single image with no features other than `Image Information`.

Filter Views:

Associated with the `debugFilter()` function.

DefaultFilterView:

Shows two images with only the basic features of `ImageInformation`, synchronized zoom and `Histogram`.

DualFilterView:

Shows the two images given to the `CVVisual` function and `Result Image` inbetween which represents the result of a filter that was applied to the others via the `Filter selection` collapsable, like a difference image between the two.

6.3 Referenz:

- Mit Hilfe von Doxygen

The screenshot shows a web browser window with the following details:

- Title Bar:** CVVisual: cvv:qutil::Accordion Class Reference
- Address Bar:** cvv mostlynerdless.de/api/classcvv_1_1qutil_1_1Accordion.html
- Content Area:**
 - Section:** Public Member Functions
 - Members:**
 - Accordion (QWidget *parent=nullptr)
 - clear ()
 - collapse (Handle handle, bool b=true)
 - collapseAll (bool b=true)
 - element (Handle handle)
 - expand (Handle handle, bool b=true)
 - expandAll (bool b=true)
 - hide (Handle handle, bool b=true)
- Sidebar:** Shows the class hierarchy and member functions for the `Accordion` class.
- Bottom Right:** Generated on Tue Mar 25 2014 22:45:17 for CVVisual by doxygen 1.8.6

7 API

7.1 Anwender API

- Triviale Benutzung auch in C++98
- Sehr klein und übersichtlich

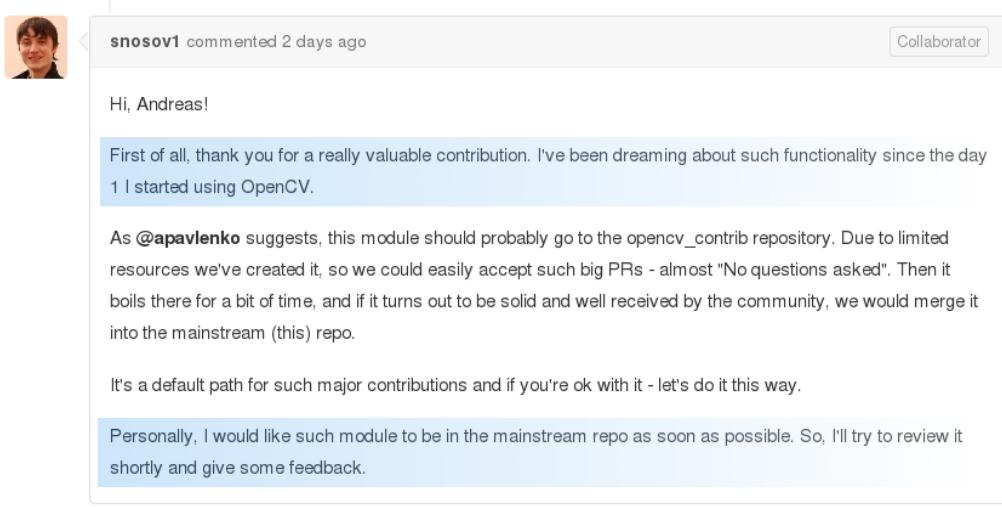
7.2 Interne API

- Erweiterung über Funktionen in `cvv::extend`
- Leichtes, zentralisiertes Hinzufügen von Visualisierungen, Filtern, Views,...

8 Ausblick

8.1 Rezeption

Projekt schien von der OpenCV-Community wohlwollend aufgenommen zu werden



8.2 Rezeption

Nach aktuellem Stand aber aufgrund C++11 und Qt5 keine Aufnahme ins Haupt-Repo

8.3 Links

- Github: <https://github.com/CVVisualPSETeam/CVVisual/>



snosov1 commented on 19. Apr.

Sorry for delay. I've looked through it right away, and they're a couple of issues. Mainly, we don't plan to enable C++11 for builds of this repository, since the support is not yet ubiquitous. Also, the usage of Qt5 is rather limiting.

This makes it a great tool for development and research on Desktops with latest sw, but is unusable on other platforms.

My thinking is that in its current form it doesn't belong to the mainstream repo because of these dependencies. But, I think, it can be merged to the contrib repo after a few minor fixes.

Let's also ask [@kirill-kornyakov](#) on that.

- Dokumentation: <https://cvv.mostlynerdless.de/>
- Doxygen: <https://cvv.mostlynerdless.de/api/>