Aufgaben zum ersten Laboratorium: Einführung in *OpenCV*

Lösen Sie die nachfolgenden Aufgaben und bereiten Sie diese bis zum nächsten Lehrveranstaltungstermin vor. Unterstrichene Aufgaben sind nach Möglichkeit während der Lehrveranstaltung zu lösen.

LB-EL 00.

a) Richten Sie *OpenCV* nach der zur Verfügung gestellten Anleitung ein und überprüfen Sie die Funktionalität der Bibliothek mit Hilfe des angegebenen Beispielprogrammes.

b) Machen Sie sich anhand der Einleitung unter `https://docs.opencv.org/4.4.0/d1/dfb/intro.html` und des Tutorials unter `http://docs.opencv.org/4.4.0/d6/d6d/tutorial_mat_the_basic_image_container.html` sowie der Beschreibung unter `http://docs.opencv.org/4.4.0/d3/d63/classcv_1_1Mat.html#details` mit der Klasse `cv::Mat` vertraut. Modifizieren Sie anschließend das in a) erstellte Beispielprogramm derart, dass es anstatt Buildinformationen auszugeben die beiden Matrizen $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ und $B = \begin{pmatrix} 4 & -3 \\ -2 & 1 \end{pmatrix}$ miteinander multipliziert und die Inverse des Ergebnisses ausgibt. Die Ausgabe soll dabei neben dem „Standardweg“ (d.h. dem überladenen <<-Operator der `cv::Mat`-Klasse) zusätzlich durch Iteration durch und Einzelzugriff auf die Matrixelemente erfolgen.

LB-EL 01.

a) Schreiben Sie ein Programm, das exakt zwei Parameter entgegennimmt, die den Dateipfad eines Eingabe- bzw. eines Ausgabebildes darstellen. Das Programm soll das Eingabebild mittels der Funktion `cv::imread` (vgl. Tutorial unter `https://docs.opencv.org/4.4.0/db/deb/tutorial_display_image.html` bzw. Beschreibung unter `http://docs.opencv.org/4.4.0/d4/da8/group__imgcodecs.html#ga288b8b3da0892bd651fce07b3bbd3a56`) in eine `cv::Mat`-Instanz einlesen, es anschließend in eine zweite `cv::Mat`-Instanz kopieren und diese mittels der Funktion `cv::imwrite` als Ausgabebild speichern. Abschließend soll das Ausgabebild mit den Funktionen `cv::imshow` und `cv::waitKey` in einem Fenster angezeigt werden, bis letzteres per Tastendruck geschlossen wird.

Benötigte Header: `opencv2/highgui.hpp`, `opencv2/imgcodecs.hpp`

b) Erweitern Sie Ihr Programm aus a) derart, dass das Ausgabebild vor dem Speichern invertiert wird, d.h. dass jedes Pixel p eines RGB-Bildes – analog zu LB-V 04. c) – auf $p' = \begin{pmatrix} v_{max} \\ v_{max} \\ v_{max} \end{pmatrix} - p$ geändert wird, wobei v_{max}

den maximal möglichen Helligkeitswert pro Farbkanal angibt. Verwenden Sie zur Umsetzung entweder `std::transform` oder `std::for_each` (**nicht** aber `cv::Mat::forEach`!), jeweils mit der `cv::Mat`-Instanz bzw. deren Iteratoren sowie einer anonymen Funktion. Sie dürfen von 8-Bit-RGB-Bildern ausgehen, d.h. der Pixeldatentyp ist `cv::Vec3b` (vgl. `https://docs.opencv.org/4.4.0/dc/d84/group__core__basic.html#ga7e6060c0b8d48459964df6e1eb524c03`). Nutzen Sie die überladenen arithmetischen Operatoren der Elternklasse, um Ihren Code einfach und lesbar zu halten. *Hinweis: Die Iteratoren der `cv::Mat`-Klasse benötigen als Templateparameter den Pixeldatentyp. Alternativ kann die `cv::Mat`-Instanz bei bekanntem Pixeldatentyp per `static_cast` auf `cv::Mat_<Pixeldatentyp>` gecastet werden, wodurch die Iteratoren den Templateparameter direkt übernehmen.*

c) Erweitern Sie Ihr Programm aus a) derart, dass vor dem Speichern des Ausgabebildes in ebendieses ein roter Kreis, ein grünes Rechteck und zwei blaue Linien eingezeichnet werden. Der Kreis soll einen Radius von 50 Pixeln haben; sein Mittelpunkt soll zudem in der Bildmitte liegen. Das Rechteck soll so beschaffen sein, dass es den Kreis so eng wie möglich umschließt, d.h., der Kreis soll wie in das Rechteck eingeschrieben aussehen. Die beiden Linien sollen das Rechteck symmetrisch in vier quadratische, gleich große Teile teilen. Alle geometrischen Figuren sollen mit einer Linienstärke von drei Pixeln und mittels der Funktionen, die im Tutorial aus `http://docs.opencv.org/4.4.0/d3/d96/tutorial_basic_geometric_drawing.html` beschrieben sind, gezeichnet werden. Zusätzlich soll mittels der Funktion `cv::putText` (vgl. Beschreibung unter `http://docs.opencv.org/4.4.0/d6/d6e/group__imgproc__draw.html#ga5126f47f883d730f633d74f07456c576`) Ihr Name nahe der Pixelposition $\begin{pmatrix} 100 \\ 100 \end{pmatrix}$ in schwarzer Schrift ausgegeben werden.

Zusätzlich benötigte Header: `opencv2/imgproc.hpp`