Software de unificación de imágenes provenientes de cámaras en microscopios ópticos

Alexandro Sifuentes Díaz
Jesús Arturo Alvarado Granadino
Instituto Tecnológico de Chihuahua II
Av. De las Industrias #11101
Complejo Ind. Chihuahua C.P. 06010,
Chihuahua, México
asdiaz@outlook.com; alvaradogranadino@yahoo.com.mx

C. Carreño-Gallardo
Centro de Investigación en Materiales Avanzados S.C.
Av. Miguel de Cervantes #120
Complejo Ind. Chihuahua C.P. 31109,
Chihuahua, México
caleb.carreno@cimav.edu.mx

Resumen: El siguiente trabajo expone el software desarrollado para periféricos auxiliares de captura de imágenes en ambientes de escritorio y/o laboratorio con un énfasis en microscopios ópticos para cubrir los problemas del procesamiento gráfico presente en el área, enfocado a la automatización de las rutinas de unificación de imágenes, problema conocido como *Image Stitching and Blending,* el cual es el conjunto de técnicas de Visión Computacional encargadas básicamente del reconocimiento de características para ordenar y posicionar un número indeterminado de imágenes generando una de mayor resolución que comprende el todo entregado en un formato asequible para el usuario del software.

También se expone la metodología propuesta, producto de la extracción y síntesis de diferentes trabajos, resumido en cuatro grandes fases: la extracción de características, empate y comparación, estimación de homografía y composición. Cada una de ellas es un grupo de algoritmos de diferentes áreas de la Visión Computacional que se detallarán a lo largo del documento.

PALABRAS CLAVE: image stitching orb open source.

1 INTRODUCCIÓN

El problema nace dentro de los laboratorios de investigación donde se utilizan dispositivos con una interfaz computadora-cámara que requieren una captura de muestras

sobre objetivos que escapan al campo de visión del lente, en este caso enfocada en la microscopía óptica.

Por una falta de diversidad de software para el microscopio, por el costo elevado de los mismos, así como por el límite computacional presente en sus equipos, se ha generado una limitante en su uso que se ve reflejado en tiempos muertos en los diferentes laboratorios donde se utilizan estos dispositivos.

Cabe aclarar que la unificación de imágenes no es un problema exclusivo de la microscopía óptica. Su aplicación se extiende a varios casos: desde la creación de imágenes en alta resolución de objetivos inmensos (como el mapeo de una ciudad), objetivos micrométricos (como la captura de una microestructura), hasta aquellas de puro carácter artístico. El objetivo principal de todos ellos es de obtener una sola imagen de gran detalle y con amplia resolución, donde la manera más óptima de conseguirlo es a partir de la unificación del conjunto.

El área computacional que investiga los algoritmos encargados de cumplir esta tarea se encuentra en las Ciencias Computacionales, en la Visión Computacional bajo el nombre de *Image Stitching and Blending* donde se estudian y buscan métodos que permitan brindar una solución rápida a este problema.

En un resumen basado en los trabajos de [1], [2], [3], [4], [5] y [6] se puede definir la metodología del sistema en las siguientes cuatro etapas.

1. **Extracción de características.** Dada una cantidad n de imágenes, se busca extraer propiedades únicas y reconocibles, utilizando diferentes alternativas como Harris Corner [7], SIFT [8], SURF [2] y ORB [6].
2. **Empate de características.** Con las características detectadas, el siguiente paso es determinar la similitud existente entre el conjunto de imágenes. Para ello se utilizan métodos como FLANN [9].
3. **Estimación de homografía.** Se estima la relación existente entre las imágenes para generar una proyección que permita la inclusión de todas ellas en un único plano.
4. **Composición.** Una vez obtenido el orden se procede al posicionamiento de las imágenes, y la corrección de problemas de exposición, movimiento y ruido para optimizar el resultado.

2 INVESTIGACIONES RELACIONADAS

Ya desde hace tiempo se han desarrollado algoritmos que permiten extraer características dentro de una imagen y no solo referente al área de *Image Stitching and Blending*, sino también como base para trabajos como el reconocimiento de patrones, el conteo de entidades, etcétera.

Los primeros se remontan a los detectores como *Moravec Corner Detector* y *Harris Corner Detector* [7] con más de veinte años de antigüedad, donde se generó la base de los algoritmos de hoy en día.

Moravec Corner Detector es uno de los primeros detectores de características y como su nombre lo indica buscaba esquinas dentro de una imagen. Su principal problema es su vulnerabilidad frente a la rotación, algo que no ocurre con el detector de Harris & Stephen [7], el cual es, cronológicamente hablando, su sustituto. Aun así, eventualmente se fueron desarrollando alternativas, principalmente porque este algoritmo presenta limitaciones frente la escalación de imágenes.

Así fue como Lowe en 1999 desarrolla SIFT [8], el cual es una de las referencias más conocidas y obvias cuando se habla de detectores de características. Su base operacional es la creación de un *Scale-Space* para obtener un conjunto de filtros de *Diferencia de Gaussianas (DoG)* que Lowe utiliza como aproximación del *Laplaciano de Gauss (LoG)*. Otra ventaja del algoritmo es que ofrece un descriptor que permite realizar el proceso de comparación y empate de características, así que, por sí sólo, cuando apareció, brindó una metodología básica completa para los problemas de *Image Stitching and Blending.*

De cualquier manera, con el paso del tiempo, fue necesario solucionar el costo computacional que implicaba su uso, pues bien, SIFT requería una carga considerable al que no cualquier computadora podía acceder y las demandas de sistemas de reconocimiento en tiempo real lo hicieron inviable.

Así, en el 2006 nace SURF [10] como una versión mejorada de SIFT, su ventaja computacional la logró mediante el uso de *Box Filters* (para la aproximación de su matriz Hessiana) y su reducción en el tamaño de su descriptor (el cual es la mitad en comparación de SIFT). Una propiedad importante de SURF es que, mientras que en SIFT se debe reducir gradualmente el tamaño de la imagen original para crear el *Scale-Space*, en SURF se pre-computan filtros de tamaño incremental. De esta manera, la imagen original no es modificada y la aplicación de los filtros sobre la misma puede ser realizada, incluso, en paralelo.

El problema con el uso de SIFT y SURF es que su uso implica la existencia de patentes, lo que requiere un pago por el desarrollo de tecnología con estos algoritmos. Para obtener una vía igual de robusta y de uso libre, en el 2011 se crea ORB [6], que, mediante la aplicación de una versión del detector FAST [11] y el descriptor BRIEF [12] ofrece una de las alternativas más poderosas en la extracción de características con resistencia a todos los factores anteriormente mencionados y que no se encuentra restringido por la existencia de patentes.

3 EXTRACCIÓN DE CARACTERÍSTICAS

El énfasis del proyecto se enfoca en esta sección. Para la selección del algoritmo óptimo se hizo un análisis sobre cuatro de los algoritmos mencionados anteriormente: Harris, SIFT, SURF y ORB.

3.1 Estudio preliminar

La extracción de características se realizó comparando la velocidad de cómputo para cada uno de los algoritmos. El resultado de dicha prueba se observa en la **Figura 1**, en ella se analizó el tiempo computacional en un total de 10 imágenes de 640×480 pixeles a color RGB (siendo éste el estándar básico de tamaño general utilizado en el área de estudio) donde se extrajo tanto las características, como sus descriptores.

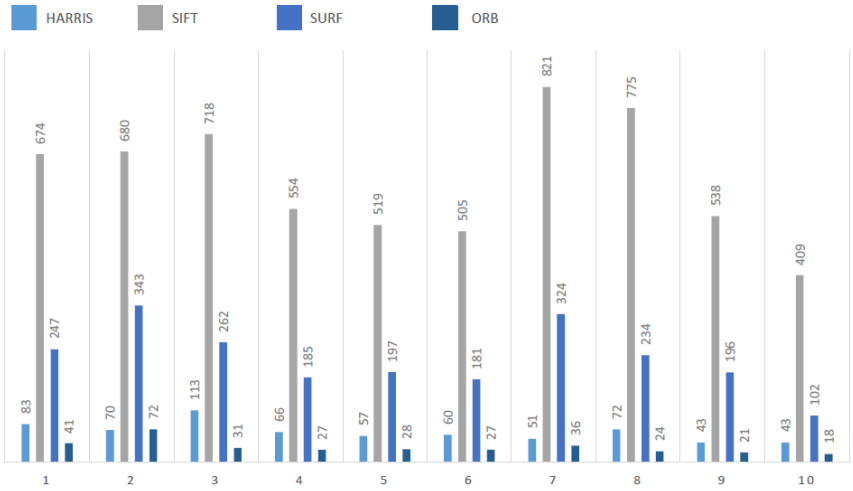

Figura 1. Tiempo computacional para los algoritmos Harris, SIFT, SURF y ORB. El resultado está expresado en milisegundos (menor es mejor).

Harris, a excepción de los otros tres, no posee un descriptor, por lo que tiene cierta ventaja de tiempo en esta prueba. Se observa, de cualquier manera, que ORB sobresalió sobre los otros algoritmos.

Otra prueba (ver **Figura 2**) fue aplicada midiendo la estabilidad de los detectores aplicando los algoritmos sobre una imagen sometida a:

- **Ruido**: Se agregó ruido gaussiano con una variación media de 10 y una desviación estándar de 10.
- **Rotación**: Se rotó la imagen original a 30^o contrarreloj.
- **Reducción**: Se redujo la imagen a un 70% del tamaño original.
- **Ampliación**: Se amplió la imagen un 30% más del tamaño original.

Los algoritmos con mejores resultados son tanto SURF como ORB. A pesar de que SIFT ofrece buenos resultados, su tiempo de cómputo le reduce practicidad.

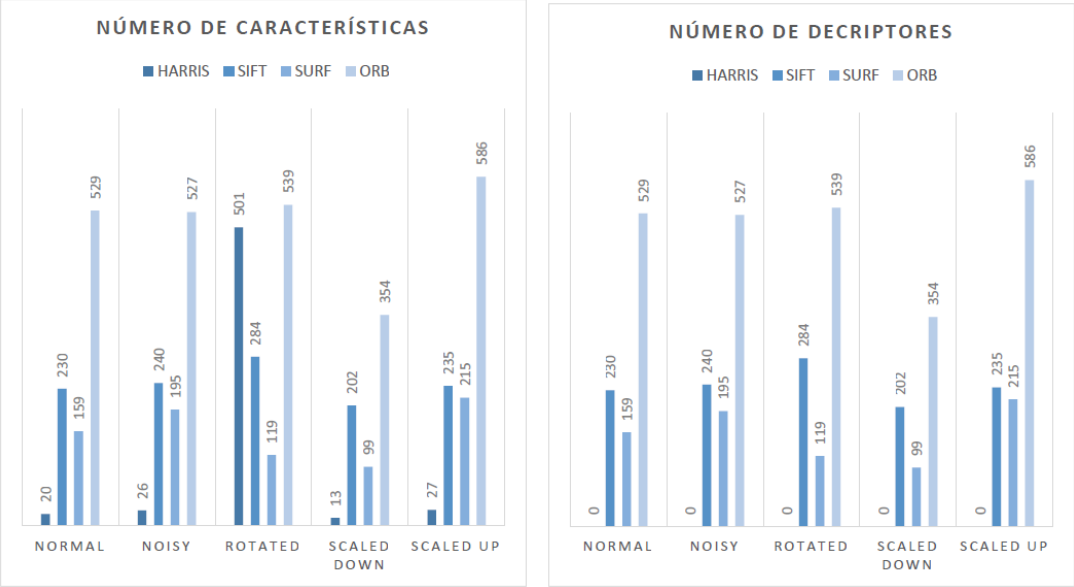

Figura 2 Tiempo computacional obtenido para imágenes distorsionadas (menor es mejor).

En base al resultado de las pruebas, se optó por utilizar ORB debido a su ventaja en su velocidad de cómputo, así también por ser un algoritmo sin restricción de patente, la explicación de su funcionamiento se abrevia en el siguiente apartado.

3.2 ORB como principal detector

Para detectar los *keypoints* (o características dentro de una imagen), ORB [6] hace uso de una versión de FAST [11], el cual mide la intensidad de un vector circular extraído para cada uno de los pixeles de la imagen a analizar con el fin de poder determinar el nivel de esquina existente. FAST no ofrece una forma de medir el keypoint, simplemente la detecta. La medición se realiza a través de una mejora incluida en ORB haciéndolo de manera similar a como lo haría el detector de Harris, donde se categoriza en relación a que tanta esquina (*cornerness*) representa el keypoint, descartando aquellos pobremente localizados.

FAST no es invariable a la escala, para atacar este problema, ORB desarrolla una pirámide de escalas, obteniendo los keypoints de los diferentes niveles dentro de la misma. La orientación del keypoint se obtiene mediante la medida de intensidad del centroide. Asumiendo que la intensidad de la esquina tiene un desplazamiento en relación al centro, este vector puede ser usado para establecer una orientación, definido como:

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{1}$$

Donde el centroide y la orientación se pueden establecer como:

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}\right) \quad \text{y} \quad \theta = atan2(m_{01}, m_{10}) \tag{2}$$

El descriptor de ORB es un vector obtenido como resultado de un conjunto de tests binarios calculados a partir del uso de imágenes integrales [15], llegando a tener un tamaño de 64 bytes (reducido en comparación de los 512 y 256 bytes de SIFT y SURF respectivamente). Otra ventaja del uso de un vector binario es que la etapa posterior de comparación y empate se realiza a través de la distancia Hamming.

Una definición formal del descriptor de ORB, basado en BRIEF, puede ser como la cadena binaria dentro de un área $\boldsymbol{p}$ de tamaño $S \times S$ llamado patch de una imagen construida a partir de test binarios τ definidos como:

$$\tau(\boldsymbol{p}, \boldsymbol{x}, \boldsymbol{y}) = \begin{cases} 1 : p(\boldsymbol{x}) < p(\boldsymbol{y}) \\ 0 : p(\boldsymbol{x}) \geq p(\boldsymbol{y}) \end{cases} \tag{4}$$

Donde $\boldsymbol{x}, \boldsymbol{y}$ son dos ubicaciones y $p(\boldsymbol{x}), p(\boldsymbol{y})$ son los valores de intensidad del pixel en la versión suavizada de $\boldsymbol{p}$ sobre el punto. De esta manera, cada conjunto de locaciones (x, y) de n_d definirán unívocamente el conjunto de test binarios sobre el patch. Pudiendo definir el descriptor como la cadena de bits:

$$f_n(\boldsymbol{p}) \coloneqq \sum_{1 \leq i \leq n} 2^{i-1} \tau(\boldsymbol{p}, \boldsymbol{x}_i, \boldsymbol{y_i}) \tag{5}$$

ORB ha optado por una distribución gaussiana en torno al patch con un vector de longitud $n = 256$. El tamaño del patch es de 31×31 pixeles con subdivisiones de 5×5. BRIEF no es un descriptor robusto frente a la rotación, para ello ORB realiza un ajuste del descriptor guiándolo con la orientación del keypoint previamente calculados. Para cada keypoint dentro del vector de longitud n en la posición (x_i, y_i) se define una matriz de $2 \times n$:

$$S = \begin{bmatrix} x_1 & \dots & x_n \\ y_1 & \dots & y_n \end{bmatrix} \tag{6}$$

Usando la orientación del patch θ y la rotación correspondiente de la matriz R_θ se construye una versión guiada S_θ de S:

$$S_\theta = R_\theta S \tag{7}$$

Modificando el operador BRIEF a:

$$g_n(p, \theta) = f_n(\boldsymbol{p}) | (x_i, y_i) \in S_\theta \tag{8}$$

En [6] se menciona la discretización de la orientación del ángulo a incrementos de $2\pi \,/\, 30 = 12^o$ y se construye una tabla de patrones precomputados de BRIEF. Esto provoca una pérdida en la variación que ORB resuelve utilizando una búsqueda codiciosa (*greedy search*) sobre todos los posibles test binario para encontrar aquellos con gran variación. Al descriptor resultante se le conoce como rBRIEF.

4 EMPATE DE CARACTERÍSTICAS

Seleccionando a ORB como el detector, el empate de características simplemente implica comparar los keypoints entre pares de imágenes para identificar la correspondencia entre estos. Todos los keypoints de una imagen son comparados con los keypoints de la otra, y el mejor resultado es identificado. Básicamente se utilizan algoritmos del vecino más cercano (k-nn) donde el uso del *vecino más cercano aproximado* (ANN) es uno de los más utilizados siendo FLANN [9] un conjunto de librerías de libre acceso que facilita ésta comparación.

Al finalizar esta etapa, se puede identificar y posicionar la localización de las imágenes para la estimación de la homografía.

5 ESTIMACIÓN DE HOMOGRAFÍA

Una vez establecido el orden de las imágenes, se debe realizar una transformación proyectiva que permita encontrar la correspondencia entre los puntos del conjunto de imágenes, esta correspondencia se le conoce como homografía y representa una matriz de 3×3 de 8 grados de libertad que contiene los parámetros de rotación, traslación y proyección. La ecuación que resuelve esta transformación es (extraída de [13, cap. 2, pág. 33]):

$$\boldsymbol{x}_i' = H\boldsymbol{x}_i \tag{9}$$

Donde $\boldsymbol{x_i}$ y $\boldsymbol{x}_i'$ son un vector de 3 elementos que representan $\boldsymbol{x} = (x, y, 1)^T$ y $\boldsymbol{x}' = (x', y', 1)^T$, donde la equivalencia con la ecuación anterior es:

$$c\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = H\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \tag{10}$$

5.1 RANSAC

Propuesto por Fischler et al [14], RANSAC es un método robusto para estimar homografías. Básicamente selecciona 4 correspondencias aleatorias y computa la homografía H. El resultado arroja un conjunto de alineaciones o desalineaciones (*inliers* y *outliers*). Este proceso es repetido iterativamente donde con cada ciclo se incrementa el número de *inliers*.

La clasificación de *inliers/outliers* es realizada en base a un umbral t donde si $\left||x' - Hx|\right| < t$, entonces es considerado un *inlier*. El número de iteraciones necesarias viene dado por la fórmula:

$$N = \log(1-p)\,/\log(1-(1-\varepsilon)^s) \tag{11}$$

Siendo
$p = probabilidad\ de\ que\ al\ menos\ una\ de\ las\ muestras\ no\ sea\ outlier.$
$\varepsilon = probabilidad\ de\ ocurrencia\ de\ un\ outlier$

Que puede ser expresado como:

$$\varepsilon = 1 - \frac{número\ de\ inliers}{número\ total\ de\ puntos} \tag{12}$$

Y $s = número\ de\ correspondencias\ usadas\ en\ cada\ iteración.$

6 COMPOSICIÓN Y APLICACIÓN

El último paso implica aplicar la transformación necesaria a partir de la homografía y del uso de técnicas de difuminado para entregar la imagen final. El desarrollo, pruebas y debug del sistema fue realizado en C++ auxiliado de las librearías de OpenCV en el entorno de desarrollo QtCreator. La **Figura 3** muestra las capturas principales del software, así como la ejecución del sistema, en ellas se aprecia el número de imágenes base, así como el resultado final obtenido.

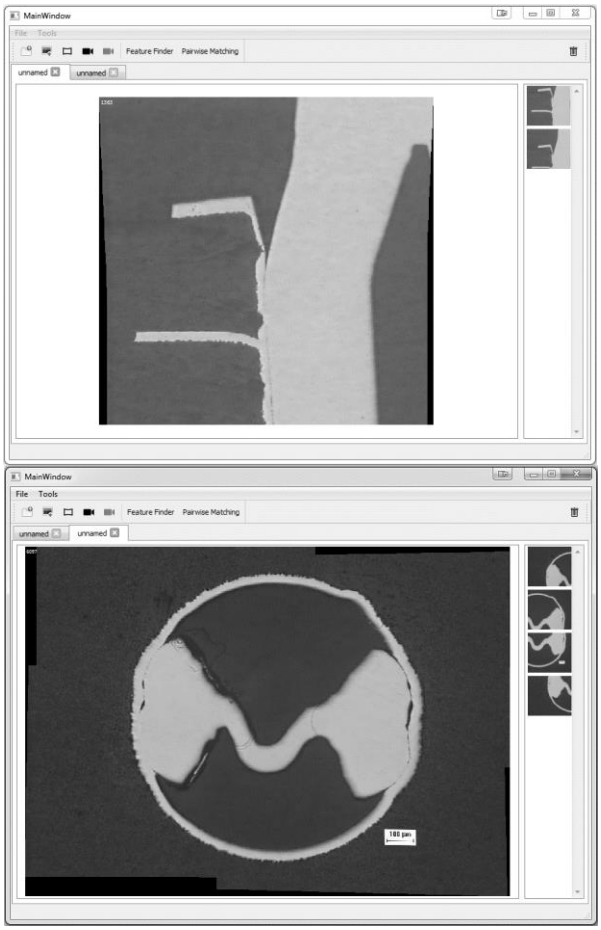

Figura 3 *Izquierda*: unión de dos imágenes a color de 640 x 480 en formato jpg, tiempo de cómputo: 1363msec. *Derecha*: unión de 4 imágenes a color de 640 x 480 en formato jpg, tiempo de cómputo: 6097msec. Se utilizó un algoritmo ORB para la extracción de características.

7 CONCLUSIONES

El resultado de la aplicación del sistema ha permitido la reducción de tiempo muerto en los equipos de cómputo. El caso de estudio dentro del centro de investigación puede ver reducido un aproximado de 2 horas a usos de 2 a 10 minutos siendo éste un tiempo estándar para la captura y unión de imágenes. Además, al ser una aplicación freeware, su uso no se encuentra restringido, por lo que cualquier interesado puede acceder al software.

Una idea importante que se manejó durante el desarrollo del sistema fue la liberación del código fuente. Se admite que la interfaz de usuario puede ser básica y que algunas

funciones pueden ser optimizadas con el tiempo, por ello, su liberación a la comunidad abre la posibilidad no solo de ofrecer una solución a los problemas de *Image Stitching*, sino también ofrece una base para futuros trabajos relacionados a ésta índole. El sistema desarrollado sigue un paradigma de ciclo de vida evolutivo bajo un esquema open-source esperando que con el tiempo, la sofisticación y adaptabilidad del mismo mejoren.

8 AGRADECIMIENTOS

Al Centro de Investigación en Materiales Avanzados S.C. por el apoyo ofrecido durante la estadía, durante la cual se facilitó el acceso a los diferentes equipos, dispositivos y recursos educativos.

REFERENCIAS

[1] M. Brown and D. G. Lowe; Automatic panoramic image stitching using invariant features. **International Journal of Computer Vision**. Vol. 74, 2007; 59-73.

[2] Herbert Bay, et al; Speeded-Up Robust Features (SURF). **Computer Vision and Image Understanding**, Elsevier Science Inc. 2008; 346-359.

[3] Krishna Paudel; Stitching of X-ray Images. PhD Thesis, UPPSALA UNIVERSITET, Institutionen för informationsteknologi, Department of Information Technology, Uppsala, 2012.

[4] Jalpa D. Mehta and S. G. Bhirud, Image Stitching Techniques. **International Conference On Contours Of Computing Technology (THINKQUEST 2010)**. Mumbai, MH, 74-80, 2010.

[5] David G. Lowe; Distinctive Image Features from Scale-Invariant Keypoints. **International Journal of Computer Vision** Vancouver, B.C., Canada. 2004; 91-110.

[6] Ethan Rublee, et al; ORB: An efficient alternative to SIFT or SURF. **ICCV '11 Proceedings of the 2011 International Conference on Computer Vision**. IEEE Computer Society; Washington, DC, USA, 2564-2571, 2011.

[7] Chris Harris and Mike Stephens; A Combined Corner and Edge Detector. **In Proc. Of Fourth Alvey Vision Conference**. 1988, 147-151.

[8] David G. Lowe; Object Recognition from Local Scale-Invariant Features. **Proceedings of the International Conference on Computer Vision ICCV '99**. IEEE Computer Society; 1999; 1150--.

[9] Marius Muja and David G. Lowe; Fast approximate nearest neighbors with automatic algorithm configuration. **In VISAPP International Conference on Computer Vision Theory and Applications**. 2009; 331-340.

[10] Herber Bay, Tinne Tuytelaars and Luc Van Gool; SURF: Speeded Up Robust Features. **ECCV (1)**. 2006; 404-417.

[11] Edward Rosten and Tom Drummond; Machine Learning for High-Speed Corner Detector. **Proceedings of the 9th European Conference on Computer Vision – Vol. 1**. Springer-Verlag; Berling, Heidelberg; 2006; 430-443.

[12] Michael Calonder et al; BRIEF: Binary Robust Independent Elementary Features. **Proceedings of the 11th European Conference on Computer Vision: Part IV**. Springer-Verlag; Berling, Heidelberg; 2010; 778-792.

[13] Richard Szeliski; Computer Vision: Algorithms and Applications. Springer; 2011.

[14] FIschler Martin A. and Bolles Robert C.; Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. **Commun. ACM**. Vol. 24, New York, NY, USA; 1981; 381-395.

[15] P. Viola and M. Jones; Rapid object detection using a boosted cascade of simple features. **CVPR 2001, Proceedings of the 2001 IEEE Computer Society Conference**. 2001; 511-518.