Software Requirements Specification for Image Feature Correspondences for Camera Calibration

Kiran Singh

2025-04-15

Contents

Revision History

Date	Version	Notes
2025-02-07	1.0	Initial Release
2025-02-27	1.1	Revision to 1.0 to address improper requirements decompisition for VnV Plan Rev. 1.0
2025-04-18	2.0	Final Release

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

Not applicable, as the project is entirely based on image processing properties in software such as pixel intensity and image resolution. The scope of this software is not to utilize properties such as camera extrinsics themselves which do use SI units for distance.

1.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with robotic vision literature. The symbols are listed in alphabetical order.

symbol	data type	description
α	32-byte int	Instance of a binary descriptor
β	32-byte int	Instance of a binary descriptor
$\mathbf{A}$	3×4 matrix of float	SE(3) transformation of the camera with respect to the target feature frame
$\mathbf{B}$	3×4 matrix of float	SE(3) transformation of the end-effector with respect to the robot-base frame
d_k	int	k^{th} bit in a binary descriptor
d_{Hamming}	int	Hamming distance computed by bitwise XOR of two binary descriptors
$\mathcal{D}_{i,j}$	$n \times 2$ int	Set of keypoint coordinates (u, v) detected for pose i and camera j
G_{2D}	$p \times p$ matrix of float	2D Gaussian kernel applied for image smoothing
i	int	Robot pose instance index
j	int	Camera instance index
$\mathcal{I}_{i,j}$	$u \times v$ matrix of int	Grayscale image captured by camera j at pose i
$I(p)$	int	Pixel intensity at location p
$I(x_i)$	int	Pixel intensity at a surrounding pixel x_i in the FAST test
k	int	Feature index within the descriptor or patch
n	int	Total number of bits in the binary descriptor (commonly 256)
$\oplus$	bitwise operator	Bitwise exclusive-OR (XOR) operation
p	int	Central pixel under test in the FAST algorithm
p_{k1}, p_{k2}	int	Sampled pixel coordinates within the descriptor patch

p'_{k1}, p'_{k2}	int	Rotated versions of p_{k1} and p_{k2} based on keypoint orientation
p_{sz}	int	Patch size used to compute BRIEF descriptors
q_k, q'_k	int	Canonical and rotated reference pixel locations used in orientation calculations
S	int	Count of surrounding pixels exceeding intensity threshold in FAST
σ	float	Standard deviation of the Gaussian kernel
t	int	Pixel intensity threshold for FAST detection
u	int	Horizontal pixel coordinate
v	int	Vertical pixel coordinate
SE(3)	4×4 matrix	Special Euclidean Group in 3D space: all rigid-body motions (rotation + translation)
$\mathbf{X}$	3×4 matrix of float	SE(3) transformation of the target feature in the robot-base frame
$\mathbf{Y}$	3×4 matrix of float	SE(3) transformation of the camera in the end-effector frame

1.3 Abbreviations and Acronyms

symbol	description
A	Assumption
BRIEF	Binary Robust Independent Elementary Features
DD	Data Definition
IFC	Image Feature Correspondences for Camera Calibration
FAST	Features from Accelerated Segment Test
FOV	Field-of-View
GD	General Definition
GS	Goal Statement
HERW	Hand-Eye Robot-World Formulation
IM	Instance Model
int	integer data type
LC	Likely Change
ORB	Oriented FAST and Rotated BRIEF
PS	Physical System Description
R	Requirement
SE(3)	Special Euclidean Group 3
SRS	Software Requirements Specification
SURF	Speeded-Up Robust Features
TM	Theoretical Model
VnV	Verification and Validation
XOR	Bitwise Exclusive-OR operation

1.4 Mathematical Notation

Unless specified otherwise, the following notation should be assumed to be the standard convention for the SRS document.

- Matrices are capitalized and are bolded, i.e. $\mathbf{X}, \mathbf{Y}$
- Column vectors are lowercase and are bolded, i.e. $\mathbf{s}, \mathbf{t}$

2 Introduction

Camera sensors are a common choice of sensor for many applications in robotics due in part to their low cost and ease of integration. Prior to their use, each camera must be calibrated such that collected data in collected imagery can be aligned with the 3D world. This process is essential to prepare the system so that imagery data can be correctly captured and processed for downstream operations.

Camera calibration consists of two aspects; intrinsic calibration and extrinsic calibration. Intrinsic calibration focuses on mapping the 2D camera image to the 3D camera frame, Extrinsic calibration converts the 3D camera frame to a global world frame. Extrinsic calibration is of significant interest as operators may need to reposition cameras on a robotic platforms for any number of operational needs.

The following section outlines the Software Requirements Specification (SRS) for a calibration algorithm that calculates the extrinsic parameters for a multi-camera robotic platform. The program is therefore named Image Feature Correspondences for Camera Calibration, or IFC.

2.1 Purpose of Document

This document is the primary resource for the user to outline the desired characteristics of the user, the required system interfaces, and desired integrated behaviour of the IFC algorithm. The goals and key assumptions of the desired software are outlined, in addition to the required definitions, theoretical models and instance models required to support its development. Specifically, theoretical models are outlined to provide a framework to promote development such that a specific design solution is not imposed at an early stage of development. The SRS is abstract - it bounds what problems need to be solved by the system, rather than how it needs to be achieved.

Following a standard waterfall development model, this document will be used as a stepping to support the development of several additional documents, each of which demonstrates successive growth in the understanding and maturation of the software product. These documents include:

1. The Design Specification: An outline of the architectural decisions that details how the requirements will be realized in the system. This is inclusive of the choice of operating environment, system interfaces with the user and its environment, and the numerical methods that shall be implemented.

2. The Verification and Validation (V&V) Plan: An outline of the specific processess to be used to assess the implementation of the code as developed from the Design Specification. Verification assessments will be used to assess whether the system has been built to the specified requirements from the SRS. Validation tests may also be outlined to ensure that the software correctly addresses the problem as defined in build confidence that the design has satisfied the outlined requirements per Section 5.

2.2 Scope of Requirements

The outlined requirements includes conventional imagery processing algorithms. When supplied with the permissible inputs, the IFC software shall scan imagery data to and identify match candidates amongst each image for various cameras and robot poses. The entire document is written under the assumption that the imagery scene is free of significant changes in ambient illumination during imagery capture.

2.3 Characteristics of Intended Reader

Readers of this document should have a introductory understanding of robot mechanics and engineering statistics, which may typically be provided in a 3rd or 4th year undergraduate course. They should also have a strong comprehension of image processing algorithms, which may consist of material covered in a 4th-year undergraduate or Master's level course in computer vision algorithms. The reader should have an understanding of robot mechanics Table 2 outlines examples of courses that satisfy the desired prerequisite knowledge criteria for the reader.

Subject	Prequisite Knowledge Level	Equivalent Course
Robot Kinematics and Dynamics	Introductory undergraduate-level	MECHENG 4K03 - Robotics
Engineering Statistics	Introductory undergraduate-level	STATS 3Y03 - Probability and Statistics for Engineering
Computer Vision Algorithms	Introductory undergraduate or graduate-level	ECE 736 - 3D Image Processing and Computer Vision

Table 2: Desired reader/user characteristics

2.4 Organization of Document

The remainder of the document uses a top-down structure that outlines, in order, the goals, assumptions, theories, definitions, and instance models. These components are then used to derive the functional and non-functional requirements. Goal statements and assumptions are sequentially distilled into theoretical models, definitions, instance models, and finally to requirements.

The reader can glean value through review of the instance models prior to the theoretical models, as the instance models outlne an operational specification of system behaviour, rather than a descriptive specification. This enables flexibility in what theoretical models are applied as they may be selected as inputs to the systems as a functional program.

3 General System Description

This section provides general information about the system. It identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints.

3.1 System Context

Figure 1 depicts the system context. A rectangle represents the IFC software itself, whereas circles depict interactions with stakeholders, namely the user. Arrowheads are used to demonstrate the sequential flow of data between the software system and the environment.

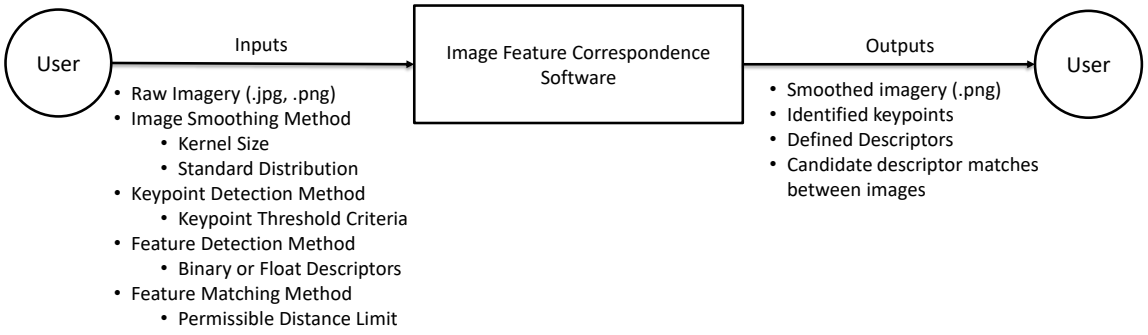


Figure 1: System Context

- User Responsibilities:
- – Provide camera images to the system
  - – Specific the desired methods of image smoothing, keypoint identification, definition of features from keypoints, and comparison of features
  - – Refine the tuning parameters used for each of the specified processes
  - – Verify that the input data is contained in the correct data structures

  - Configure the type of feature-handler algorithms to be used with corresponding threshold criteria

- Image Feature Correspondences for Camera Calibration Responsibilities:
  - Detect data type mismatch, such as a string of characters instead of a floating point number
  - Assess whether inputs constitute a fully defined physical setup
  - Calculate match candidates between two images
  - Provide a quality metric of confidence in the calculated outputs

This software is not intended for use in safety-critical applications. If it is required for applications that are deemed to be safety-critical, then the software will need to undergo a new review and verification cycle.

3.2 User Characteristics

As the intent of this software is intended to be used primarily in a research environment, the intended user is expected to posses a similar prerequisite knowledge base as that of the intended reader of the SRS, as outlined in Section 2.3. Therefore, the user of the IFC software should be familiar with the following prerequisite material outlined in Table 2 prior to use of the software.

3.3 System Constraints

The IFC software shall be compatible with Python 3.1 libraries, such as OpenCV.

4 Specific System Description

This section first presents the problem description, which gives a high-level view of the problem to be solved. This is followed by the solution characteristics specification, which presents the assumptions, theories, definitions and finally the instance models.

4.1 Problem Description

Image Feature Correspondences for Camera Calibration is intended to evaluate how imagery data from robot-based cameras can can be manipulated to define and align features between separate images in support of downstream operations for extrinsic camera calibration.

4.1.1 Terminology and Definitions

This subsection provides a list of terms that are used in the subsequent sections and their meaning, with the purpose of reducing ambiguity and making it easier to correctly understand the requirements:

- **Features:** distinctive patterns or structures in an image that are identifiable and useful for matching between images
- **Keypoints:** specific pixel locations in an image that represent significant and repeatable features
- **Correspondences:** pairs of keypoints between two images that represent represent the same real-world point
- **Extrinsic Parameters:** the transform between the 3D camera frame to the 3D world frame.
- **Intrinsic Parameters:** camera parameters that pertain to the transform of the 2D image plane frame to the 3D camera frame
- **Hand-eye:** the relation between the robot end-effector to the camera frame
- **Robot-world:** the relation between the robot base frame to the world frame
- **Pose:** refers to the position and orientation of an object, sensor, or robot in $SE(3)$
- **Patch:** a region of defined size ($p_{sz} \times p_{sz}$) that is used to compare pixels and define a feature descriptor

4.1.2 Physical System Description

PS1: Frames for the robot base, hand, camera, and target landmark.
PS2: Known base-to-hand transform.
PS3: Projected camera images from distinct camera poses.

The physical system of Image Feature Correspondences for Camera Calibration, as shown in Figure 2, outlines case of a single-camera, single target configuration (PS1). This outlines the frames of interest for for the broader problem formulation.

The single-camera configuration can be extrapolated to the multi-camera, multi-target case (PS2), as shown in Figure 3, from which a representation of equivalent keypoints between images (PS3) is shown in Figure 4. The formulation in Figure 4 is what will be addressed by the IFC software.

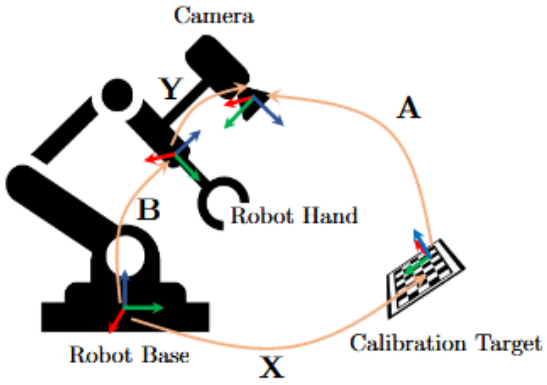


Figure 2: PS1: Single-camera robotic manipulator robot-world hand eye configuration. Modified from Wang et al. (2022).

.

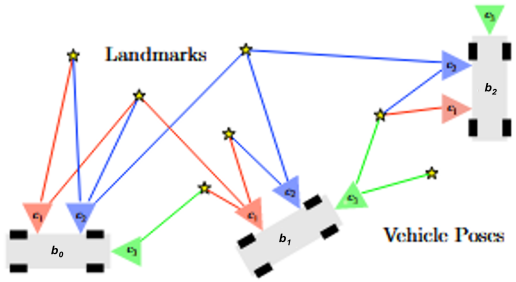


Figure 3: PS2: Multi-camera mobile robotic platform. Modified from Kaveti et al. (2024).

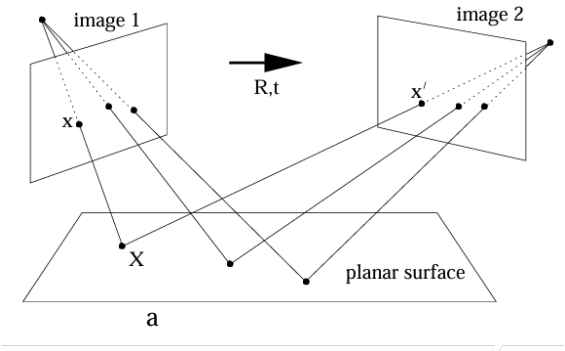


Figure 4: PS3: Multi-camera mobile robotic platform. Modified from Hartley and Zisserman (2000).

4.1.3 Goal Statements

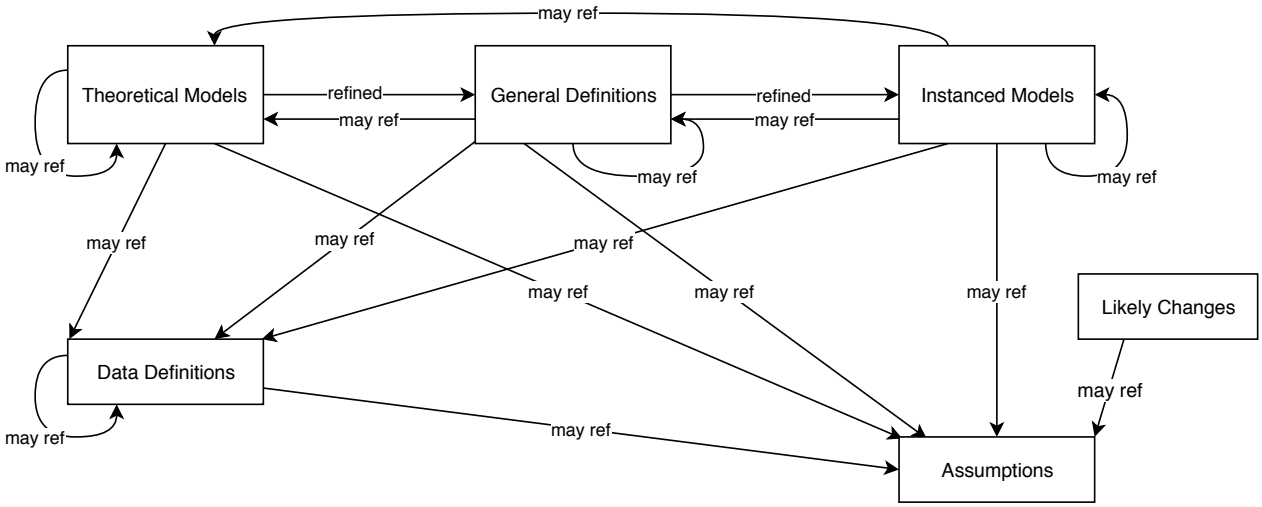
The system goals follow.

GS1: Identify a collection of features from each input image frame.

GS2: Identify feature correspondences between all input images.

GS3: Generate a report of the identified feature correspondences.

4.2 Solution Characteristics Specification



The instance models that govern Image Feature Correspondences for Camera Calibration are presented in Section 4.2.8. The information to understand the meaning of the instance models and their derivation is also presented, so that the instance models can be verified.

4.2.1 Types

Omitted for Rev 1.0 release.

4.2.2 Scope Decisions

Control of the ambient illumination conditions falls outside the scope of this software. It is the responsibility of the user to verify that the ambient lighting conditions do not change to a significant degree during the image capture process.

4.2.3 Modelling Decisions

Omitted for Rev 1.0 release.

4.2.4 Assumptions

This section simplifies the original problem and helps in developing the theoretical model by filling in the missing information for the physical system. The numbers given in the square brackets refer to the theoretical model [TM], general definition [GD], data definition [DD], instance model [IM], or likely change [LC], in which the respective assumption is used.

A1: Imagery shall be provided by at least one camera (GD4).

A2: All supplied imagery is produced by a pinhole model, affine camera (TM2, , GD1).

A3: All imagery will be input as greyscale data (GD1).

A4: The solution is not limited to memory constraints observed in hardware used for real-time applications (TM3).

4.2.5 Theoretical Models

This section introduces the general equations and laws that are used to build the Image Feature Correspondences for Camera Calibration software.

Number	TM1
Label	**N-Dimensional Gaussian Kernel**
Equation	$G_{\text{N-Dim}}(x, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\|\vec{x}\|^2}{2\sigma^2}\right)$
Description	$G_{\text{N-Dim}}$ represents the probability density function of an n-dimensional Gaussian distribution. $\vec{x}$ is the n-dimensional input vector, and σ is the standard deviation controlling the spread of the distribution. This kernel is often used in smoothing, weighting, and noise modeling.
Notes	All variables are unitless.
Source	Chung (2007)
Ref. By	GD1
Pre-conditions for TM1:	None
Derivation for TM1:	None

Number	TM2
Label	**Features from Accelerated Segment Test (FAST)**
Equation	$S = \sum_{i \in x} (\lvert I(p) - I(x_i) \rvert > t) > 12$
Description	For a 2D image acquired via a pinhole affine camera model (A2), p denotes the central pixel of interest, and $x = \{x_1, \ldots, x_{16}\}$ are the 16 contiguous pixels forming a circle around p. $I(p)$ and $I(x_i)$ are the pixel intensities at locations p and x_i, respectively. t is a user-defined intensity threshold. The segment test S counts how many surrounding pixels differ from p by more than t. If more than 12 such pixels are found, p is identified as a keypoint.
Notes	All variables are unitless.
Source	Rosten and Drummond (2006)
Ref. By	GD2, LC1
Pre-conditions for TM2:	None
Derivation for TM2:	None

Number	TM3
Label	**Binary Robust Independent Elementary Features (BRIEF)**
Equation	$\mathbf{d} = \{d_1, d_2, \ldots, d_{256}\}, \quad \text{where } d_k = \begin{cases} 1 & \text{if } I(p_{k1}) < I(p_{k2}) \\ 0 & \text{otherwise} \end{cases}$
Description	$\mathbf{d}$ is a 256-bit (32-byte) binary descriptor vector. To compute each bit d_k, a square patch of size $p_{sz} \times p_{sz}$ is centered at a detected keypoint. Two pixels, p_{k1} and p_{k2}, are randomly sampled from this patch. If the intensity at p_{k1} is less than that at p_{k2}, d_k is set to 1; otherwise, it is set to 0.
Notes	BRIEF is efficient but may not be suitable for real-time applications under strict memory constraints (A4).
Source	OpenCV Contributors (2024)
Ref. By	GD3, LC2
Pre-conditions for TM3:	None
Derivation for TM3:	None

Number	TM4
Label	**Hamming Distance**
Equation	$d_{\text{Hamming}}(\mathbf{d}_\alpha, \mathbf{d}_\beta) = \sum_{i=0}^{n-1} (d_{\alpha,i} \oplus d_{\beta,i})$
Description	The Hamming distance measures the number of differing bits between two binary descriptors, $\mathbf{d}_\alpha$ and $\mathbf{d}_\beta$. Each descriptor is composed of n bits. The bitwise exclusive-OR operation ($\oplus$) is used to compare corresponding bits. The total number of differing bits is returned as an integer. This is not a geometric distance, but a metric in Hamming space.
Notes	All variables are unitless.
Source	OpenCV (2021)
Ref. By	IM4, LC3
Pre-conditions for TM4:	None
Derivation for TM4:	None

4.2.6 General Definitions

This section collects the laws and equations that will be used in building the instance models.

Number	GD1
Label	**2-Dimensional Gaussian Kernel**
SI Units	Unitless
Equation	$G_{2D}(u, v, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{u^2+v^2}{2\sigma^2}\right)$
Description	G_{2D} represents the 2D Gaussian kernel function applied to a grayscale image (A2, A3). The variables u and v denote the horizontal and vertical distances from the kernel center, respectively (both are unitless). The parameter σ represents the standard deviation of the Gaussian distribution, which controls the amount of smoothing applied. This kernel is convolved with the image to reduce high-frequency noise and enhance regional uniformity.
Source	TM1
Ref. By	IM1

Number	GD2
Label	**FAST Implementation**
Units	Unitless
Equation	$\sum_{k \in x} (\lvert I_k - I(u, v) \rvert > t) \geq N$
Description	The threshold count of $x \in \{1 \dots 16\}$ is concretely defined as N. $I(u, v)$ represents the pixel intensity of pixel p. I_k represents the grayscale image intensity of the k^{th} pixel in the circle around pixel p. t represents the user-defined threshold to define an allowable range of pixel intensity.
Source	TM2
Ref. By	IM1

Number	GD3
Label	**Rotated BRIEF**
Units	Unitless
Equation	$d_k = \begin{cases} 1 & \text{if } I(p'_{k1}) < I(p'_{k2}) \\ 0 & \text{otherwise} \end{cases}$
Description	d_k represents the k^{th} bit in the BRIEF descriptor. For each bit, two points p'_{k1} and p'_{k2} are selected within a patch centered at the keypoint. These point pairs are generated by rotating a canonical sampling pattern according to the keypoint's orientation. This process ensures that BRIEF descriptors are invariant to in-plane rotations.
Source	OpenCV Contributors (2024)
Ref. By	IM3

Detailed derivation of Rotated Brief Transform

p_k and q_k are calculated by successive increments of a 12°. The transform between $p_k^{'}$ and $q_k^{'}$ is outlined below, where θ represents the increase in rotation by a denomination of 12°.

$$\begin{bmatrix} p'_k \\ q'_k \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} p_k \\ q_k \end{bmatrix}$$

4.2.7 Data Definitions

No data definitions are outlined for the current release.

4.2.8 Instance Models

This section transforms the problem defined in Section 4.1 into one which is expressed in mathematical terms. It uses refined derivations of the models outlined in Sections 4.2.5 and 4.2.6.

Number	IM1
Label	**Gaussian-Smoothed Image,** $\mathcal{I}'_{i,j}(u, v, \sigma)$
Input	$\mathcal{I}_{i,j}(u, v)$, the raw grayscale image from TM1 $G_{2D}(x, y, \sigma)$, the 2D Gaussian kernel from GD1
Output	$\mathcal{I}'_{i,j}(u, v, \sigma)$, the Gaussian-smoothed image
Equation	$\mathcal{I}'_{i,j}(u, v, \sigma) = \sum_{x=-k}^{k} \sum_{y=-k}^{k} G_{2D}(x, y, \sigma) \cdot \mathcal{I}_{i,j}(u - x, v - y)$
Description	$\mathcal{I}'_{i,j}(u, v, \sigma)$ is the output image obtained by convolving the input image $\mathcal{I}_{i,j}(u, v)$ with a Gaussian kernel $G_{2D}(x, y, \sigma)$ of size $(2k + 1) \times (2k + 1)$. The kernel size is selected based on the standard deviation σ, and defines the smoothing window centered at pixel (u, v). The convolution suppresses high-frequency components (e.g., noise), producing a smoother version of the input image.
Sources	Chung (2007)
Ref. By	GS1, GS2, IM2

Number	IM2
Label	**Image Keypoint Detection,** $\mathcal{D}_{i,j}(u, v)$
Input	$\mathcal{I}_{i,j}(u, v, \sigma)$ from IM1, intensity threshold t
Output	$\mathcal{D}_{i,j}(u, v)$ — a set of keypoints, each defined by pixel coordinates (u, v) in the image plane.
Description	$\mathcal{D}_{i,j}(u, v)$ represents the set of image keypoints detected in the smoothed grayscale image $\mathcal{I}_{i,j}(u, v, \sigma)$ using the FAST algorithm described in TM2. Each keypoint corresponds to a pixel location (u, v) for which the segment test S exceeds a count of 12, indicating a corner-like intensity pattern around that pixel. t is the user-defined pixel intensity threshold that determines contrast sensitivity.
Sources	Rosten and Drummond (2006)
Ref. By	GS1, IM3

Number	IM3
Label	**Produce Feature Descriptors,** D_{bin}
Input	$\mathcal{D}_{i,j}(u,v)$ from IM2, p_{sz}, l_{bin}
Output	D_{bin}, a variable-length vector of binary feature descriptors
Description	$\mathcal{D}_{i,j}(u,v)$ is the set of keypoints defined by their pixel coordinates, as obtained from IM2. p_{sz} is the size of the square pixel patch (in pixels) used to compute each descriptor. l_{bin} specifies the number of binary comparisons (i.e., bits) to generate per descriptor, as required by TM3. D_{bin} is the resulting collection of BRIEF descriptors, each computed from randomly sampled pixel pairs within a patch centered on a keypoint.
Sources	OpenCV Contributors (2024)
Ref. By	GS2, IM4

Number	IM4
Label	**Inter-Image Descriptor Comparison,** $dist_{Hamming}$
Input	d_a, d_b from IM3
Output	$dist_{Hamming}$, an integer value
Description	d_a and d_b are binary descriptors generated from different keypoints, as defined in IM3. $dist_{Hamming}$ is computed as the Hamming distance between these descriptors, equal to the number of differing bits. Each bit-wise comparison uses the exclusive-OR operator ($\oplus$), as formalized in TM4.
Sources	OpenCV (2021)
Ref. By	GS2

4.2.9 Input Data Constraints

Table 3 shows the data constraints on the input output variables. The column for physical constraints gives the physical limitations on the range of values that can be taken by the variable. The column for software constraints restricts the range of inputs to reasonable values. The software constraints will be helpful in the design stage for picking suitable algorithms. The constraints are conservative, to give the user of the model the flexibility to experiment with unusual situations. The column of typical values is intended to provide a feel for a common scenario. The uncertainty column provides an estimate of the confidence with which the physical quantities can be measured. This information would be part of the input if one were performing an uncertainty quantification exercise.

The specification parameters are outlined in Table 3.

Table 3: Input Variables

Var	Physical Constraints	Software Constraints	Typical Value	Uncertainty
$\mathcal{I}_{i,j}(u,v)$	N/A	$0 \leq \mathcal{I}_{i,j}(u,v) \leq 255$	N/A	N/A
σ	$\sigma > 0$	$\sigma > 0$	0.05	N/A

(*) $\mathcal{I}_{i,j}(u,v)$ is limited to being an 8-bit unsigned integer

(*) standard deviation, σ, is constrained to be positive and real

4.2.10 Properties of a Correct Solution

Each image needs to be assessed such that it is invariant to the order for which it is processed by the software. In other words, the output of the Image Feature Correspondences for Camera Calibration software should be deterministic. Each identified keypoint needs to have an associated set of pixel coordinates. Each descriptor needs to have an associated keypoint and unique identifier. Each candidate match between features needs to have associated keypoints, descriptors, and identifiers for the points of origin of both consituitive images.

As it stands, there are currently no identified physical constraints on the system.

5 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

5.1 Functional Requirements

R1: The IFC software shall accept updates to the variance in image noise, σ, upon user input.

R2: The IFC software shall accept updates to the image intensity threshold, t, upon user input.

R3: The IFC software shall accept updates to the patch size, p_s, upon user input.

R4: The IFC software shall accept updates to the bin size, l_{bin}, upon user input.

R5: The IFC software shall use a default noise suppresion parameter of $\sigma = 1$ if no option is specified by the user.

R6: The IFC software shall use corner detection as the default feature detection method if no option is specified by the user.

R7: The IFC software shall use binary descriptors as the default keypoint descriptor method if no option is specified by the user.

R8: The IFC software shall use compare feature descriptors by their Hamming Distance if no alternative descriptor matching method is specified by the user.

R9: The IFC software shall implement noise reduction on an input greyscale image per a prescribed standard deviation (from IM1).

R10: The IFC software shall, given a 2D greyscale image, define a set of keypoints per the alloted threshold criteria (from IM2).

R11: The IFC software shall define feature descriptors from identified keypoints (from IM3).

R12: The IFC software shall identify matches between descriptors that originate from separate images (from IM4).

R13: The IFC software shall review all identified feature matches to confirm that they originate from separate images.

R14: The IFC software shall report that all feature correspondences are uniquely defined by two respective sets of cameras and pose frames as an output.

R15: The IFC software shall report the identified matches between features, with corresponding camera and pose identifiers, as an output.

5.2 Nonfunctional Requirements

NFR1: **Reliability** The IFC software shall be invariant to the order that imagery data is provided to the feature matching process.

NFR2: **Usability** The IFC system shall provide an interface that at least 80% of its users would describe as 'simple to use' based on direct feedback.

NFR3: **Maintainability** The IFC software should support implementation of new feature detection methods within 30% of the total time effort required to develop the default method of feature detection.

NFR4: **Performance** The IFC software should report timing metrics for each image processing operation.

NFR5: **Performance** The system should report the average and maximum memory used during each operation.

5.3 Rationale

The scope and assumptions of this document are intended to encapsulate a realistic problem formulation software. This software is intended for use in laboratory experimentation and configurations, rather than as an off-the-shelf product for applications that involve more risk, such as those that are safety-critical or mission critical in scope.

A1 is necessary at it enables flexibility to developers to scale the software to the case of multiple cameras in the event that the use case of a single camera can be successfully implemented. However, it does not specify that support for multiple cameras is necessary. A2 is necessary as all imagery needs to be projected on a 2D affine plane as the software is not intended to be used with imagery that is procurred using a fish-eye lens. A3 is necessary as it significant reduces the complexity of computation that is required to process colour imagery. Conversion to greyscale is a standard industrial practice for image processing. A4 is necessary to avoid imposing timing and performance constraints on the software. This would consitute the need to perform a hazard analysis to address safety and performance challenges that would arise from operating in a real-time scenario.

6 Likely Changes

LC1: The theoretical model of keypoint detection will be abstracted to expand the types of implementation models that can be used.

LC2: The theoretical model of assigning feature descriptors will be abstracted to expand the types of implementation models that can be used.

LC3: The theoretical model of descriptor matching will be abstracted to expand the types of implementation models that can be used.

7 Unlikely Changes

LC4: It is unlikely that the design will need to be adjusted to perform computation on RGB data instead of grayscale data.

8 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 6 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 7 shows the dependencies of instance models, requirements, and data constraints on each other. Table 5 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

	A1	A2	A3	A4
TM1			X	
TM2		X		
TM3		X		X
TM4	X			
GD1		X		
GD2				
GD3				
IM1				
IM2				
IM3				
IM4				
LC1				
LC2				
LC3				

Table 5: Traceability Matrix Showing the Connections Between Assumptions and Other Items

	TM1	TM2	TM3	TM4	GD1	GD2	GD3	IM1	IM2	IM3	IM4	LC1	LC2	LC3
TM1					X			X						
TM2						X			X			X		
TM3							X			X			X	
TM4											X			X
GD1								X						
GD2									X					
GD3										X				
IM1									X					
IM2										X				
IM3											X			
IM4														
LC1						X			X					
LC2							X			X				
LC3											X			

Table 6: Traceability Matrix Showing the Connections Between Items of Different Sections

	IM1	IM2	IM3	IM4	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15
IM1		X			X				X				X						
IM2			X			X				X				X					
IM3				X			X	X			X				X				
IM4												X				X	X	X	X
R1	X												X						
R2		X												X					
R3			X												X				
R4			X												X				
R5													X						
R6	X													X					
R7		X													X				
R8			X													X			
R9	X												X						
R10		X																	
R11			X																
R12				X															
R13				X															
R14				X															
R15		X	X	X															

Table 7: Traceability Matrix Showing the Connections Between Requirements and Instance Models

9 Development Plan

An outline of the expected development objectives and milestones is outlined in the VnV Plan.

10 Values of Auxiliary Constants

No auxillary constants are defined for this software.

References

M. K. Chung. The gaussian kernel. https://pages.stat.wisc.edu/~mchung/teaching/MIA/reading/diffusion.gaussian.kernel.pdf, 2007. [Online; accessed 2-Feb-2025].

Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, USA, 2000. ISBN 0521623049.

Pushyami Kaveti, Matthew Giamou, Hanumant Singh, and David M. Rosen. Oasis: Optimal arrangements for sensing in slam. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13818–13824, 2024. doi: 10.1109/ICRA57147.2024.10611644.

OpenCV. Feature matching with flann, 2021. URL https://docs.opencv.org/3.4/d5/d6f/tutorial_feature_flann_matcher.html. Accessed: 2025-02-06.

OpenCV. Fast algorithm for corner detection, 2025. URL https://docs.opencv.org/3.4/df/d0c/tutorial_py_fast.html. Accessed: 2025-04-14.

OpenCV Contributors. Orb (oriented fast and rotated brief) - opencv tutorial, 2024. URL https://docs.opencv.org/3.4/d1/d89/tutorial_py_orb.html. Accessed: 2025-02-05.

Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, pages 430–443, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-33833-8.

Yifu Wang, Wenqing Jiang, Kun Huang, Sören Schwertfeger, and Laurent Kneip. Accurate calibration of multi-perspective cameras from a generalization of the hand-eye constraint. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 1244–1250, 2022. doi: 10.1109/ICRA46639.2022.9811577.