Software Requirements Specification for Image Feature Correspondences for Camera Calibration: subtitle describing software

Kiran Singh

2025-02-07

Contents

Revision History

Date	Version	Notes
2022-02-05	1.0	Initial Release

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

Throughout this document SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	unit	SI
m	length	metre
s	time	second

[Only include the units that your SRS actually uses. —TPLT]

[Derived units, like newtons, pascal, etc, should show their derivation (the units they are derived from) if their constituent units are in the table of units (that is, if the units they are derived from are used in the document). For instance, the derivation of pascals as $\mathrm{Pa} = \mathrm{N\,m^{-2}}$ is shown if newtons and m are both in the table. The derivations of newtons would not be shown if kg and s are not both in the table. —TPLT]

[The symbol for units named after people use capital letters, but the name of the unit itself uses lower case. For instance, pascals use the symbol Pa, watts use the symbol W, teslas use the symbol T, newtons use the symbol N, etc. The one exception to this is degree Celsius. Details on writing metric units can be found on the NIST web-page. —TPLT]

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 the heat transfer literature and with existing documentation for solar water heating systems. The symbols are listed in alphabetical order.

symbol	unit	description
i	Unitless	Robot pose instance
j	Unitless	Camera Instance
k	Unitless	Target Feature Instance
$\mathbf{A}$	3×4 Matrix	SE(3) Equivalent transformation of a camera with respect to the target feature frame
$\mathbf{B}$	3×4 Matrix	SE(3) Equivalent transformation of the end-effector in the robot-base frame
$\mathbf{R}$	3×3 Matrix	Rotation matrix in SO(3)

t	3×3 Matrix	Translation matrix in SE(3)
$\mathbf{X}$	3×4 Matrix	SE(3) Equivalent transformation of the target feature in the robot-base frame
$\mathbf{Y}$	3×4 Matrix	SE(3) Equivalent transformation of the camera in the end-effector frame

[Use your problems actual symbols. The si package is a good idea to use for units. —TPLT]

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
LC	Likely Change
ORB	Oriented FAST and Rotated BRIEF
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
SURF	Speeded-Up Robust Features
TM	Theoretical Model
V&V	Verification and Validation

[Add any other abbreviations or acronyms that you add —TPLT]

1.4 Mathematical Notation

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

- Scalars are lowercase and are not bolded, i.e. a, b

[This section is optional, but should be included for projects that make use of notation to convey mathematical information. For instance, if typographic conventions (like bold face font) are used to distinguish matrices, this should be stated here. If symbols are used to show mathematical operations, these should be summarized here. In some cases the easiest way to summarize the notation is to point to a text or other source that explains the notation. —TPLT]

[This section was added to the template because some students use very domain specific notation. This notation will not be readily understandable to people outside of your domain. It should be explained. —TPLT]

[Guiding principles for the SRS document:

- Do not repeat the same information at the same abstraction level. If information is repeated, the repetition should be at a different abstraction level. For instance, there will be overlap between the scope section and the assumptions, but the scope section will not go into as much detail as the assumptions section.

—TPLT]

[When the documentation is done, it should be possible to trace back to the source of every piece of information. Some information will come from external sources, like terminology. Other information will be derived, like General Definitions. —TPLT]

[An SRS document should have the following qualities: unambiguous, consistent, complete, validatable, abstract and traceable. —TPLT]

[The overall goal of the SRS is that someone that meets the Characteristics of the Intended Reader (Section 2.3) can learn, understand and verify the captured domain knowledge. They should not have to trust the authors of the SRS on any statements. They should be able to independently verify/derive every statement made. —TPLT]

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 may be refered to as Image Feature Correspondences, 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 4.1.

2.2 Scope of Requirements

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

[Modelling the real world requires simplification. The full complexity of the actual physics, chemistry, biology is too much for existing models, and for existing computational solution techniques. Rather than say what is in the scope, it is usually easier to say what is not. You can think of it as the scope is initially everything, and then it is constrained to create the actual scope. For instance, the problem can be restricted to 2 dimensions, or it can ignore the effect of temperature (or pressure) on the material properties, etc. —TPLT]

[The scope section is related to the assumptions section (Section 4.2.4). However, the scope and the assumptions are not at the same level of abstraction. The scope is at a high level. The focus is on the "big picture" assumptions. The assumptions section lists, and describes, all of the assumptions. —TPLT]

[The scope section is relevant for later determining typical values of inputs. The scope should make it clear what inputs are reasonable to expect. This is a distinction between scope and context (context is a later section). Scope affects the inputs while context affects how the software will be used. —TPLT]

2.3 Characteristics of Intended Reader

Reviewers of this document should have a , and a strong understanding of image processing algorithms. A 4th year undergraduate or Master's level course in Computer Vision algorithm is strongly recommended. The reviewer should have an understanding of robot mechanics per a 3rd or 4th year undergraduate course. The developer should also have a general background in statistics and functional programming.

[This section summarizes the skills and knowledge of the readers of the SRS. It does NOT have the same purpose as the "User Characteristics" section (Section 3.2). The intended readers are the people that will read, review and maintain the SRS. They are the people that will conceivably design the software that is intended to meet the requirements. The user, on the other hand, is the person that uses the software that is built. They may never read this SRS document. Of course, the same person could be a "user" and an "intended reader." —TPLT]

[The intended reader characteristics should be written as unambiguously and as specifically as possible. Rather than say, the user should have an understanding of physics, say what kind of physics and at what level. For instance, is high school physics adequate, or should the reader have had a graduate course on advanced quantum mechanics? —TPLT]

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.

[This section provides a roadmap of the SRS document. It will help the reader orient themselves. It will provide direction that will help them select which sections they want to read, and in what order. This section will be similar between project. —TPLT]

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.

[The purpose of this section is to provide general information about the system so the specific requirements in the next section will be easier to understand. The general system description section is designed to be changeable independent of changes to the functional requirements documented in the specific system description. The general system description provides a context for a family of related models. The general description can stay the same, while specific details are changed between family members. —TPLT]

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.

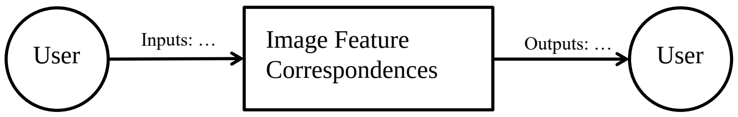


Figure 1: System Context

[For each of the entities in the system context diagram its responsibilities should be listed. Whenever possible the system should check for data quality, but for some cases the user will need to assume that responsibility. The list of responsibilities should be about the inputs and outputs only, and they should be abstract. Details should not be presented here. However, the information should not be so abstract as to just say "inputs" and "outputs". A summarizing phrase can be used to characterize the inputs. For instance, saying "material properties" provides some information, but it stays away from the detail of listing every required properties. —TPLT]

- User Responsibilities:
  - Provide input data to the system
  - Verify that the input data uses the correct units
  - 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 define physical setup
  - Calculate required outputs

[Identify in what context the software will typically be used. Is it for exploration? education? engineering work? scientific work?. Identify whether it will be used for mission-critical or safety-critical applications. —TPLT] [This additional context information is needed to determine how much effort should be devoted to the rationale section. If the application is safety-critical, the bar is higher. This is currently less structured, but analogous to, the idea to the Automotive Safety Integrity Levels (ASILs) that McSCert uses in their automotive hazard analyses. —TPLT]

3.2 User Characteristics

The end user of the IFC software should have an undergraduate understanding of introductory mechanicas of robots, statistics, and computer vision algorithms.

[This section summarizes the knowledge/skills expected of the user. Measuring usability, which is often a required non-function requirement, requires knowledge of a typical user. As mentioned above, the user is a different role from the "intended reader," as given in Section 2.3. As in Section 2.3, the user characteristics should be specific an unambiguous. For instance, "The end user of Image Feature Correspondences for Camera Calibration should have an understanding of undergraduate Level 1 Calculus and Physics." —TPLT]

3.3 System Constraints

[System constraints differ from other type of requirements because they limit the developers' options in the system design and they identify how the eventual system must fit into the world. This is the only place in the SRS where design decisions can be specified. That is, the quality requirement for abstraction is relaxed here. However, system constraints should only be included if they are truly required. —TPLT]

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 of the 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 within a given reference frame.
- Patch: A square region of an image of an assigned size.

4.1.2 Physical System Description

The physical system of Image Feature Correspondences for Camera Calibration, as shown in Figure 2, outlines case of a single-camera, single target configuration. This outlines the frames of interest for 4.1.2. This configuration can be extrapolated to the multi-camera, multi-target case, as shown in 4.1.2. A representation of equivalent keypoints between images is shown in 4.1.2.

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 purpose of this section is to clearly and unambiguously state the physical system that is to be modelled. Effective problem solving requires a logical and organized approach. The statements on the physical system to be studied should cover enough information to solve the problem. The physical description involves element identification, where elements are defined as independent and separable items of the physical system. Some example elements include acceleration due to gravity, the mass of an object, and the size and shape of an object. Each element should be identified and labelled, with their interesting properties specified clearly. The physical description can also include interactions of the elements, such as the following: i) the interactions between the elements and their physical environment; ii) the interactions between elements; and, iii) the initial or boundary conditions. —TPLT]

[The elements of the physical system do not have to correspond to an actual physical entity. They can be conceptual. This is particularly important when the documentation is for a numerical method. —TPLT]

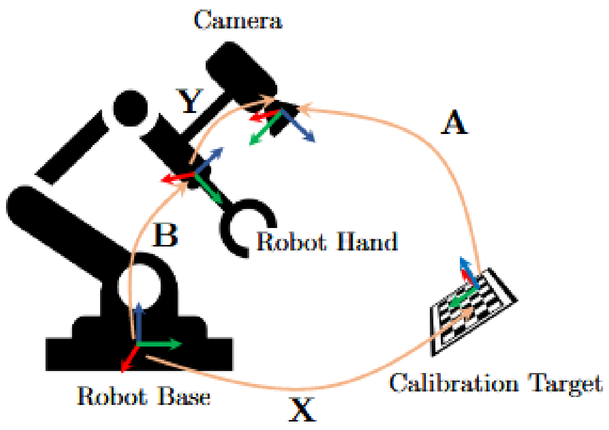


Figure 2: Single-camera robotic manipulator robot-world hand eye configuration. Modified from Wang et al. (2022).

.

4.1.3 Goal Statements

[The goal statements refine the "Problem Description" (Section 4.1). A goal is a functional objective the system under consideration should achieve. Goals provide criteria for sufficient completeness of a requirements specification and for requirements pertinence. Goals will be refined in Section "Instanced Models" (Section 4.2.8). Large and complex goals should be decomposed into smaller sub-goals. The goals are written abstractly, with a minimal amount of technical language. They should be understandable by non-domain experts. —TPLT]
Given the [inputs —TPLT], the goal statements are:

GS1: Define the method(s) used to search for feature correspondences between each image frame.

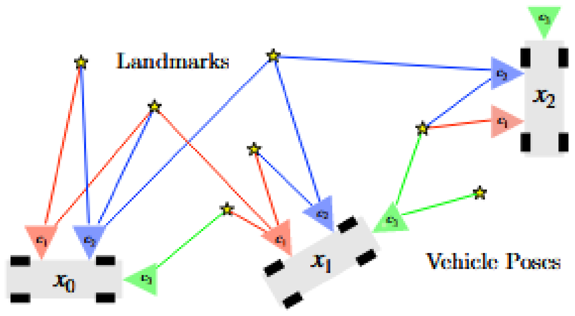


Figure 3: Multi-camera mobile robotic platform. Modified from Kaveti et al. (2024).

.

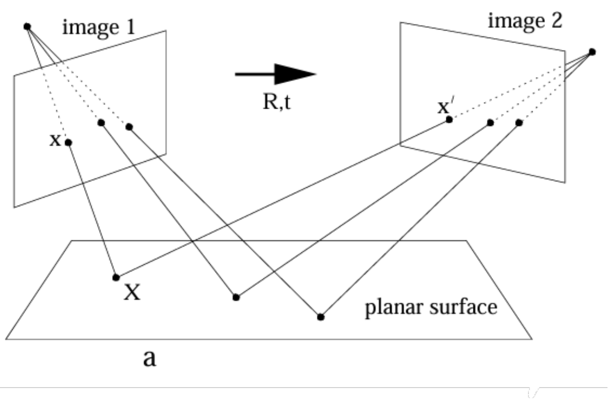


Figure 4: Multi-camera mobile robotic platform. Modified from Hartley and Zisserman (2000).

.

GS2: Define the method(s) used to search for feature correspondences between each image frame.

GS3: Identify a collection of features from each image frame.

GS4: Identify feature correspondences between images.

GS5: Generate a report of the identified feature correspondences.

4.2 Solution Characteristics Specification

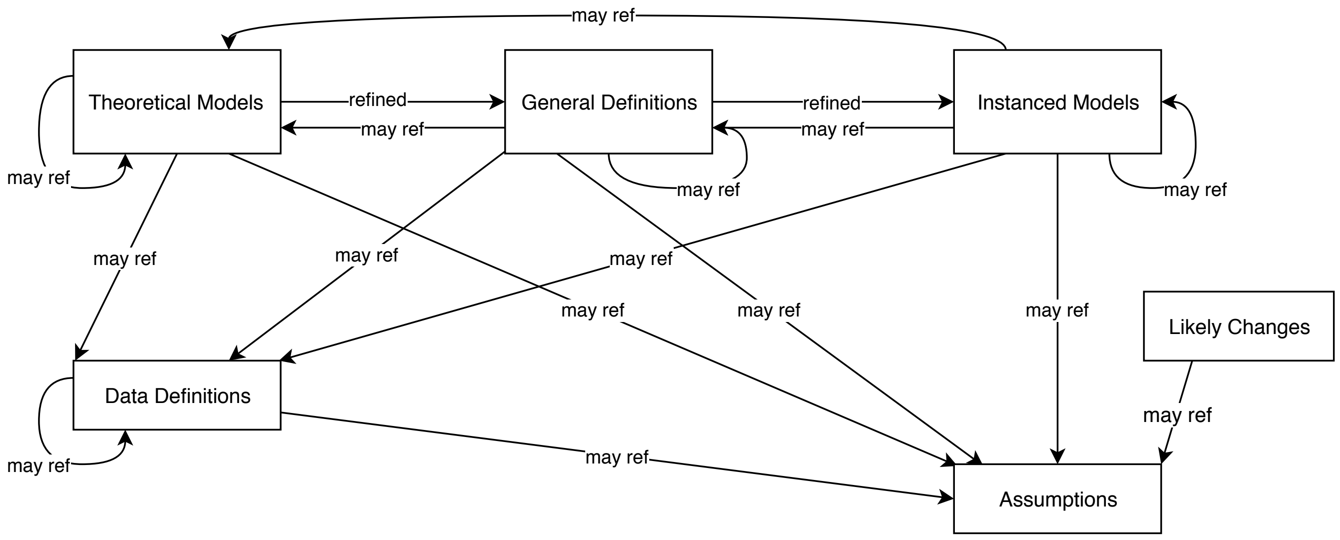
[This section specifies the information in the solution domain of the system to be developed. This section is intended to express what is required in such a way that analysts and stakeholders get a clear picture, and the latter will accept it. The purpose of this section is to reduce the problem into one expressed in mathematical terms. Mathematical expertise is used to extract the essentials from the underlying physical description of the problem, and to collect and substantiate all physical data pertinent to the problem. —TPLT]

[This section presents the solution characteristics by successively refining models. It starts with the abstract/general Theoretical Models (TMs) and refines them to the concrete/specific Instance Models (IMs). If necessary there are intermediate refinements to General Definitions (GDs). All of these refinements can potentially use Assumptions (A) and Data Definitions (DD). TMs are refined to create new models, that are called GMs or IMs. DDs are not refined; they are just used. GDs and IMs are derived, or refined, from other models. DDs are not derived; they are just given. TMs are also just given, but they are refined, not used. If a potential DD includes a derivation, then that means it is refining other models, which would make it a GD or an IM. —TPLT]

[The above makes a distinction between "refined" and "used." A model is refined to another model if it is changed by the refinement. When we change a general 3D equation to a 2D equation, we are making a refinement, by applying the assumption that the third dimension does not matter. If we use a definition, like the definition of density, we aren't refining, or changing that definition, we are just using it. —TPLT]

[The same information can be a TM in one problem and a DD in another. It is about how the information is used. In one problem the definition of acceleration can be a TM, in another it would be a DD. —TPLT]

[There is repetition between the information given in the different chunks (TM, GDs etc) with other information in the document. For instance, the meaning of the symbols, the units etc are repeated. This is so that the chunks can stand on their own when being read by a reviewer/user. It also facilitates reuse of the models in a different context. —TPLT]
[The relationships between the parts of the document are show in the following figure. In this diagram "may ref" has the same role as "uses" above. The figure adds "Likely Changes," which are able to reference (use) Assumptions. —TPLT]



The instance models that govern Image Feature Correspondences for Camera Calibration are presented in Subsection 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

[This section is optional. Defining types can make the document easier to understand. —TPLT]

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

[This section is optional. —TPLT]

4.2.4 Assumptions

[If it helps with the organization and understandability, the assumptions can be presented as sub sections. The following sub-sections are options: background theory assumptions, helper theory assumptions, generic theory assumptions, problem specific assumptions, and rationale assumptions —TPLT]

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 GD1, GD2, GD3.

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

[Theoretical models are sets of abstract mathematical equations or axioms for solving the problem described in Section “Physical System Description” (Section 4.1.2). Examples of theoretical models are physical laws, constitutive equations, relevant conversion factors, etc. —TPLT]

[Optionally the theory section could be divided into subsections to provide more structure and improve understandability and reusability. Potential subsections include the following:

Context theories, background theories, helper theories, generic theories, problem specific theories, final theories and rationale theories. —TPLT]

This section focuses on the general equations and laws that Image Feature Correspondences for Camera Calibration is based on. [Modify the examples below for your problem, and add additional models as appropriate. —TPLT]

Number	TM1
Label	**N-Dimensional Gaussian Kernel**
Equation	$G_{N-Dim}(\overrightarrow{x},\sigma) = \frac{1}{2\pi\sigma^2}e^{-\frac{\overrightarrow{x}}{2\sigma^2}}$
Description	The above equation represents the distribution of data when defined as a n-dimensional Gaussian Distribution. $\overrightarrow{x}$ is the collection of n-dimensions in the Gaussian distribution. σ is the standard deviation of the Gaussian distribution.
Notes	All variables are unitless.
Source	Chung
Ref. By	GD1
Pre-conditions for TM1:	None
Derivation for TM1:	None

Number	TM2
Label	**Features from Accelerated Segment Test (FAST)**
Equation	$S_{p\to x} = \begin{cases} d, , I_{p\to x} \leq I_p - t & \text{(darker)} \\ s, , I_{p\to x} \leq I_p + t & \text{(similar)} \\ b, , I_p + t \leq I_{p\to x} & \text{(brighter)} \end{cases}$
Description	x represents the 16 contiguous pixels that surround pixel p, or $x \in \{1, ..., 16\}$. $S_{p\to x}$ represents the comparison of pixel intensity for p to x, with an assignment of d, s, or b to represent that pixel p is brighter, darker, or similar to its neighbours. If the pixel is classified as either a b or a d, then the pixel is defined as a keypoint for edge detection.
Notes	All variables are unitless.
Source	Rosten and Drummond (2006)
Ref. By	GD2, LC**??**
Pre-conditions for TM2:	None
Derivation for TM2:	None

Number	TM3
Label	**Binary Robust Independent Elementary Features (BRIEF)**
Equation	$d_k = \mathbf{1}(I(p_k) < I(q_k))$
Description	For a selected patch within an image, and d_k represents the kth bit in a descriptor, pixels p_k and q_k are randomly sampled.
Notes	This theory is computationally expensive and may not be suited to real-time applications (A4).
Source	OpenCV Contributors (2024)
Ref. By	GD3, LC**??**
Pre-conditions for TM3:	None
Derivation for TM3:	None

Number	TM4
Label	**Hamming Distance**
Equation	$d_{Hamming}(a,b) = \sum_{i=0}^{n-1}(a_i \bigoplus b_i)$
Description	The Hamming distance is used to compare two binary numbers by each succesive bit, and return the sum, $d_{Hamming}$, for the binary descriptors a and b. i represents the index within the total of n descriptors. Both descriptors are assumed to originate from separate images (A1).
Notes	All variables are unitless.
Source	OpenCV (2021)
Ref. By	IM4, LC3
Pre-conditions for TM4:	None
Derivation for TM4:	None

[“Ref. By” is used repeatedly with the different types of information. This stands for Referenced By. It means that the models, definitions and assumptions listed reference the current model, definition or assumption. This information is given for traceability. Ref. By provides a pointer in the opposite direction to what we commonly do. You still need to have a reference in the other direction pointing to the current model, definition or assumption. As an example, if TM1 is referenced by GD2, that means that GD2 will explicitly include a reference to TM1. —TPLT]

4.2.6 General Definitions

[General Definitions (GDs) are a refinement of one or more TMs, and/or of other GDs. The GDs are less abstract than the TMs. Generally the reduction in abstraction is possible through invoking (using/referencing) Assumptions. For instance, the TM could be Newton’s Law of Cooling stated abstracting. The GD could take the general law and apply it to get a 1D equation. —TPLT]

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

[Some projects may not have any content for this section, but the section heading should be kept. —TPLT] [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Number	GD1
Label	**2-Dimensional Gaussian Kernel**
SI Units	Unitless
Equation	$G_{2D}(u, v, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$
Description	G_{2D} represents the gaussian kernel transform that is applied to the 2D greyscale image (A2, A3) given by horizonal pixel u and vertical pixel v, both of which are unitless. σ represents the allowable standard deviation of the two dimensional distribution for a given image. The Gaussian kernel is used to smooth an image to reduce noise amongst the individual pixels. $\overrightarrow{x}$ is the collection of n-dimensions in the Gaussian distribution. σ is the standard deviation of the Gaussian distribution.
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 $x \in \{1 \ldots 16\}$ 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 for the planar image(A2). 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	$d_k = I(p_k^{'}) < I(q_k^{'})$
Description	For a defined patch size around a selected keypoint, pixels p and q are randomly selected. d_k represents the kth bit in a descriptor, pixels $p_k^{'}$ and $q_k^{'}$ are randomly sampled. This ensures that BRIEF can be implemented in a manner that is rotation invariant to an image within a plane.
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

[The Data Definitions are definitions of symbols and equations that are given for the problem. They are not derived; they are simply used by other models. For instance, if a problem depends on density, there may be a data definition for the equation defining density. The DDs are given information that you can use in your other modules. —TPLT]

[All Data Definitions should be used (referenced) by at least one other model. —TPLT]

This section collects and defines all the data needed to build the instance models. The dimension of each quantity is also given. [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Number	DD1
Label	**2D Gaussian Kernel**
Symbol	G_{2D}
SI Units	Unitless
Equation	$G_{2D}(x, y, \sigma) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$
Description	G_{2D} represents the gaussian kernel transform that is ultimately applied to 2D image given by horizonal pixel x and vertical pixel y, both of which are unitless. σ represents the allowable standard deviation of the two dimensional distribution for a given image. The Gaussian kernel is used to smooth an image to reduce noise amongst the individual pixels.
Sources	Chung
Ref. By	IM??

4.2.8 Instance Models

[The motivation for this section is to reduce the problem defined in “Physical System Description” (Section 4.1.2) to one expressed in mathematical terms. The IMs are built by refining the TMs and/or GDs. This section should remain abstract. The SRS should specify the requirements without considering the implementation. —TPLT]

This section transforms the problem defined in Section 4.1 into one which is expressed in mathematical terms. It uses concrete symbols defined in Section 4.2.7 to replace the abstract symbols in the models identified in Sections 4.2.5 and 4.2.6.

The goals [reference your goals —TPLT] are solved by [reference your instance models —TPLT]. [other details, with cross-references where appropriate. —TPLT] [Modify the examples below for your problem, and add additional models as appropriate. —TPLT]

Number	IM1
Label	**Gaussian-Smoothed Image,** $\mathcal{I'}_{i,j}(u, v, \sigma)$
Input	$\mathcal{I}_{i,j}(u, v)$, σ from GD1
Output	$\mathcal{I'}_{i,j}(u, v, \sigma)$ from TM1
Description	$\mathcal{I'}_{i,j}(u, v, \sigma)$ is the smoothed output image. $\mathcal{I}_{i,j}(u, v, \sigma)$ is the unfiltered input image. σ is the standard deviation of the 2D Gaussian Distribution.
Sources	Chung
Ref. By	GS3, GS4,IM2

Number	IM2
Label	**Image Keypoint Detection,** $\mathcal{D}_{i,j}(u, v)$
Input	$\mathcal{I}_{i,j}(u, v, \sigma)$ from IM1, t The input is constrained so that $T_{\text{init}} \leq T_C$ (A**??**)
Output	$\mathcal{D}_{i,j}(u, v)$, such that D is an n-dimensional 2D matrix of the horizontal and vertical pixel coordinates in the form of (u,v).
Description	$\mathcal{D}_{i,j}(u, v)$ defined by the mapping transform, $S_{p \to x}$, as outlined in TM2. $\mathcal{I}_{i,j}(u, v, \sigma)$ is the smoothed image (m x n) from IM1. t is the pixel intensity threshold.
Sources	Rosten and Drummond (2006)
Ref. By	GS1, GS3, IM3

Number	IM3
Label	**Produce Feature Descriptors,** D_{bin}
Input	$\mathcal{D}_{i,j}(u, v)$ from IM2, p_s, l_{bin}
Output	D_{bin} as as variable size vector
Description	$\mathcal{D}_{i,j}$ is the vector of pixels (u,v) of identified keypoints, per IM2. p_s is the size of pixel patch to draw samples, as an integer. l_{bin} is the size of the binary descriptor, in bits, per TM3. D_{bin} is the output vector of identified descriptors, per TM3.
Sources	OpenCV Contributors (2024)
Ref. By	GS2, GS4, IM4

Number	IM4
Label	**Inter-Image Descriptor Comparison,** $dist_{Hamming}$
Input	d_a, d_b from IM3
Output	$dist_{Hamming}$
Description	$dist_{Hamming}$ is the sum of bitwise **XOR** binary descriptors as outlined in TM3. d_a, d_b are binary descriptors as defined by IM3.
Sources	OpenCV (2021)
Ref. By	GS4, GS5

4.2.9 Input Data Constraints

Table 2 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 in Table 2 are listed in Table **??**.

(*) [you might need to add some notes or clarifications —TPLT]

Table 2: Input Variables

Var	Physical Constraints	Software Constraints	Typical Value	Uncertainty
m	$m > 0$	$m > 0$	N/A	N/A
n	$n > 0$	$n > 0$	N/A	N/A
σ	$\sigma > 0$	$\sigma > 0$	N/A	N/A
$p(u,v)$	N/A	$0 \leq p(u,v) \geq 255$	N/A	N/A

4.2.10 Properties of a Correct Solution

A correct solution must exhibit high recall. Recall can be described by the comparison of true positive (TP) over the total predicted results, which is the sum of all true positive and false negative (FN) outcomes. A specific performance metric may be assigned [LC5000].

$$Recall = \frac{TP}{TP + FN}$$

This table should be addressed for a subsequent revision. As it standard, there are currently no identified physical constraints on the system.

Table 4: Output Variables

Var	Physical Constraints

5 Requirements

[The requirements refine the goal statement. They will make heavy use of references to the instance models. —TPLT]

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 the default noise suppresion parameters if no option is specified by the user.

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

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

R8: The IFC software shall use the default descriptor matching method if no option 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 provide a .csv file that outlines the respective feature matches.

R14: The IFC software shall verify that all features are matched to features that originate from separate images.

R15: The IFC software shall verify that all feature correspondences are uniquely defined by the camera of origin, image frame of origin, and their unique pixels.

R16: The IFC software shall verify that all feature correspondences are uniquely defined by the two respective sets of cameras, image frame, and associated pixels.

[Every IM should map to at least one requirement, but not every requirement has to map to a corresponding IM. —TPLT]

5.2 Nonfunctional Requirements

[List your nonfunctional requirements. You may consider using a fit criterion to make them verifiable. —TPLT] [The goal is for the nonfunctional requirements to be unambiguous, abstract and verifiable. This isn't easy to show succinctly, so a good strategy may be to give a "high level" view of the requirement, but allow for the details to be covered in the Verification and Validation document. —TPLT] [An absolute requirement on a quality of the system is rarely needed. For instance, an accuracy of 0.0101 % is likely fine, even if the requirement is for 0.01 % accuracy. Therefore, the emphasis will often be more on describing now well the quality is achieved, through experimentation, and possibly theory, rather than meeting some bar that was defined a priori. —TPLT] [You do not need an entry for correctness in your NFRs. The purpose of the SRS is to record the requirements that need to be satisfied for correctness. Any statement of correctness would just be redundant. Rather than discuss correctness, you can characterize how far away from the correct (true) solution you are allowed to be. This is discussed under accuracy. —TPLT]

NFR1: **Usability** The IFC software shall be compatible with Python 3.1 libraries, such as OpenCV.

5.3 Rationale

[Provide a rationale for the decisions made in the documentation. Rationale should be provided for scope decisions, modelling decisions, assumptions and typical values. —TPLT]

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 8 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

[You will have to modify these tables for your problem. —TPLT]

[The traceability matrix is not generally symmetric. If GD1 uses A1, that means that GD1's derivation or presentation requires invocation of A1. A1 does not use GD1. A1 is "used by" GD1. —TPLT]

[The traceability matrix is challenging to maintain manually. Please do your best. In the future tools (like Drasil) will make this much easier. —TPLT]

	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

The purpose of the traceability graphs is also to provide easy references on what has to be additionally modified if a certain component is changed. The arrows in the graphs represent dependencies. The component at the tail of an arrow is depended on by the component at the head of that arrow. Therefore, if a component is changed, the components that it points to should also be changed. Figure **??** shows the dependencies of theoretical models, general

	IM1	IM2	IM3	IM4	4.2.9	R1	R2	R3	R4	R5	R5	R6	R6	R7	R8	R9	R10
IM1																	
IM2																	
IM3																	
IM4																	
4.2.9																	
R1																	
R2																	
R3																	
R4																	
R5																	
R6																	
R7																	
R8																	
R9																	
R10																	
R11																	
R12																	
R13																	
R14																	
R15																	
R16																	

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

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

Table 8: Traceability Matrix Showing the Connections Between Assumptions and Other Items

definitions, data definitions, instance models, likely changes, and assumptions on each other. Figure **??** shows the dependencies of instance models, requirements, and data constraints on each other.

9 Development Plan

[This section is optional. It is used to explain the plan for developing the software. In particular, this section gives a list of the order in which the requirements will be implemented. In the context of a course this is where you can indicate which requirements will be implemented as part of the course, and which will be "faked" as future work. This section can be organized as a prioritized list of requirements, or it could should the requirements that will be implemented for "phase 1", "phase 2", etc. —TPLT]

10 Values of Auxiliary Constants

[Show the values of the symbolic parameters introduced in the report. —TPLT]

[The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance. —TPLT]

[The value of FRACTION, for the Maintainability NFR would be given here. —TPLT]

References

M. K. Chung. The gaussian kernel. https://pages.stat.wisc.edu/~mchung/teaching/MIA/reading/diffusion.gaussian.kernel.pdf. [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 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.

End of Document.