الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سكيكدة

Faculty of Science

Department of Computer Science

Field : Mathematics and Computer Science

Major : Computer Science

Option : Information Systems and Software Engineering

Final Year Project Report for the Completion of Master Degree

in Computer Science

Title

Video Stitching

Prepared by:

DJAREDDIR Kawther

Supervised by:

TOUIL GHASSEN

Academic year: 2024-2025

DEDICATION

With deep gratitude, I dedicate this humble work to:

- *My beloved parents, who have never spared any effort for me and have always supported me and stood by my side.*
- *My dear sister Rayan and brother Ayoub.*
- *My supervisor Mr Touil Ghassen.*
- *I also thank my grandmothers Djamila and Fatma, and my grandfathers Ali and Rabah, may God have mercy on them.*

Djareddir Kawther

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Mr TOUIL Ghassen, for he's unwavering support, invaluable guidance, and expert knowledge throughout the entire research process. His dedication and mentorship have been instrumental in shaping the direction and quality of this project.

I would also like to express my heartfelt thanks to Professor BOUTINE RACHID, my first supervisor, for his initial guidance and support in the early stages of this research. Despite facing health challenges "شفاه الله و عافاه و البسه "لباس الصحة و العافية

He continued to provide invaluable assistance, and his contributions have been crucial to the development of this project. His support has been deeply appreciated, and I am grateful for the foundation he helped establish for my work.

Also, I am deeply thankful to the faculty members of the Computer Science department at the University of 20 Aout- 1955- SKIKDA for their insightful courses, enlightening discussions, and the intellectual environment they provided. Their expertise and passion for their respective fields have been a constant source of inspiration for me.

Abstract

This project explores video stitching—a process that merges multiple video streams into a single panoramic output. Building on image stitching techniques like feature detection, homography estimation, and blending, it addresses video-specific challenges such as temporal consistency, motion handling, and real-time processing. A Python-based application was developed using OpenCV and PyQt6, offering both simple and advanced stitching modes. Advanced methods include hybrid stitching, attention-based blending, and GAN-based inpainting. Evaluations show that while fast methods are efficient, advanced techniques produce superior visual quality, making the system suitable for applications in VR, surveillance, and mapping.

Table of Content

Chapter II : Video Stitching

Chapter III : Implementation

List of Figures

List of Tables

List of Diagrams

General Introduction

In a world increasingly pervaded by visual stimuli, the ability to understand, reconstruct, and extend visual scenes has become a central goal in a wide range of fields, from computer vision and robotics to immersive media and smart surveillance systems. Digital imaging devices are increasingly expected to deliver wide, continuous, and immersive fields of view, whether to document environments, inform decision-making, or enhance user experience.

Yet the physical constraints of sensor sizes and camera optics impose a restriction on how much of the world can be captured in a single shot. To mitigate this issue, image and video stitching technologies have been implemented. These technologies merge overlapping views be these photographs or video frames—into continuous and expanded depictions of a single scene.

The process of image stitching, or the compositing of static images, has infiltrated applications like panoramic photography, mapping, and 360° media; nonetheless, this is merely the fundamental aspect of an even greater challenge: video stitching. In video stitching, the objective is to effectively merge multiple streams of video that have been captured from a variety of viewpoints, frequently in real time. These comprise not only geometric coincidence but also frame synchronization, motion control, temporal coherence maintenance, and dynamic object control that are part of the scene.

Video stitching is now essential in many applications, including virtual reality, autonomous navigation, surveillance, and immersive broadcasting. It enables the generation of 360° videos, real-time panoramic views, and dense spatial awareness. Seamless video stitching, however, demands sophisticated techniques, including motion tracking, stabilization, temporal blending, and typically learning-based models.

This dissertation presents a comprehensive review of video stitching. It explores general principles, methods, and issues with panoramic video system design without dwelling on any particular subproblem. It aims to get the entire pipeline and emphasize the technical underpinnings that make video stitching technologies possible today.

CHAPTER I

Introduction To Image Stitching

1. Historical Background

Computer vision is a branch of artificial intelligence that enables machines to interpret and understand visual data from the world, replicating tasks that require human visual perception. It encompasses techniques such as image enhancement, object recognition, motion tracking, and 3D scene reconstruction. One of its longstanding goals is to expand visual understanding beyond the limitations of a single image precisely what image and video stitching aim to achieve. The development of stitching technologies closely mirrors the broader evolution of computer vision: from handcrafted methods to automated, learning-based systems.

The history of image stitching dates back to the early days of photography in the 19th century, when photographers manually combined multiple prints to create panoramic views, such as the iconic "Great Picture" of San Francisco (1878), assembled from 30 individual images. However, the formal computational foundations of modern image stitching emerged in the 1980s and 1990s with advancements in computer vision and projective geometry. Early algorithms relied on simple techniques like cross-correlation for alignment, but the field transformed with the introduction of invariant feature detectors, such as Scale-Invariant Feature Transform (SIFT) by David Lowe in 1999, which enabled robust matching of keypoints across images with varying scales and orientations.

This breakthrough was followed by accelerated alternatives like Speeded-Up Robust Features (SURF) in 2006 and Oriented FAST and Rotated BRIEF (ORB) in 2011, which traded slight accuracy for real-time performance. Parallel developments in homography estimation and bundle adjustment allowed seamless warping of images into a common plane, while blending techniques like multi-band blending (e.g., Burt and Adelson’s Laplacian pyramids in 1983) minimized visible seams. The rise of consumer applications—such as Apple’s PhotoStitch (2009) and Google’s Street View—popularized automated stitching, and deep learning later revolutionized the field with end-to-end neural networks (e.g., DeepHomography in 2016) that bypassed traditional feature extraction. Today, image stitching is a mature yet evolving discipline, bridging computational photography, medical imaging, and satellite photogrammetry, while its principles underpin the more complex challenges of video stitching [13].

Table 1: Key Milestones in the Evolution of Image and Video Stitching.

Era	Year	Milestone / Technique	Significance
Early Photography	1843	First known panorama (Joseph Puchberger)	Manual stitching of daguerreotypes for wide-field views.
Pre-Computational	1878	"The Great Picture" of San Francisco	30+ photos manually assembled into a panorama.
Algorithmic Foundations	1983	Laplacian Pyramid Blending (Burt & Adelson)	Introduced multi-band blending to minimize seams.
	1994	Autostitch (early software)	First automated tools for photo stitching.
Feature-Based Revolution	1999	SIFT (Scale-Invariant Feature Transform)	Robust feature matching across scales/rotations.
	2006	SURF (Speeded-Up Robust Features)	Faster alternative to SIFT with similar accuracy.
	2011	ORB (Oriented FAST and Rotated BRIEF)	Real-time feature detection for mobile devices.
Consumer Adoption	2009	Apple PhotoStitch (iOS)	Brought stitching to mainstream photography.
	2010s	Google Street View & Photo Sphere	Large-scale automated stitching for mapping/Virtual Reality.
Deep Learning Era	2016	Deep Homography (Nguyen et al.)	First CNN-based homography estimation, bypassing feature extraction.
	2018+	GAN-based blending (e.g., SPHP-GAN)	Improved seam removal using generative adversarial networks.
Present & Future	2020s	Real-time neural stitching (e.g., RNNs for video)	Transition from images to video stitching with temporal consistency.

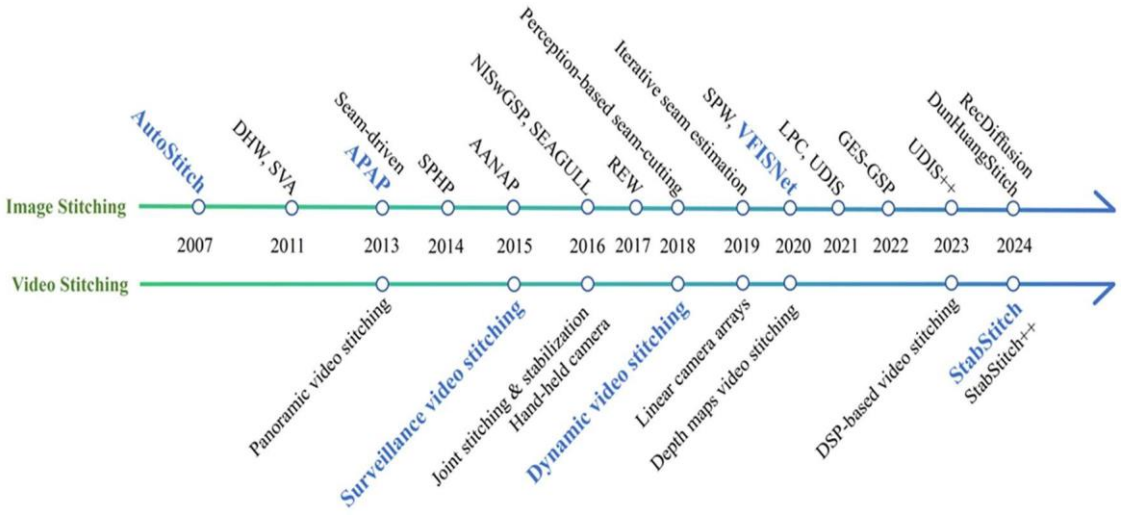

Figure 1: The timeline of the development of image and video stitching lists some representative algorithms. The top is image stitching, the bottom is video stitching, and the blue bold font is the turning point of development.

2. Digital Image and Video Representations: Foundations for Stitching

Having explored the historical evolution and algorithmic advancements in image stitching, it is now essential to understand the digital content upon which these techniques operate. At the core of both image and video stitching lie pixel-based visual data still images and moving frames. A solid grasp of their structure, representation formats, and key characteristics forms the foundation for understanding how stitching algorithms manipulate, align, and blend visual information.

This section introduces the two primary forms of digital visual data: still images (digital photographs) and moving images (digital video). It begins by examining still image formats, particularly raster representations, which are central to stitching due to their pixel-level detail. The discussion then transitions to digital video, which extends the concept of stitching to time-sequenced imagery and introduces temporal complexities unique to video processing.

2.1. Still Image

In engineering and computer science, still images must be represented in a format that allows computational manipulation. This is achieved through digital images, which are numerical representations of static two-dimensional scenes. The conversion from analog to

digital, known as digitization, is typically performed by scanners or digital cameras. Once digitized, these images are ready for processing and analysis [13].

Digital images generally fall into two main categories: raster (bitmap) and vector formats. Raster images are composed of a grid of pixels, each carrying specific color or intensity information. Vector images, by contrast, represent shapes and lines mathematically using points and curves.

In the context of image stitching, raster images are the primary focus. This is because stitching algorithms operate at the pixel level detecting features, aligning regions, and blending overlapping areas based on pixel data.

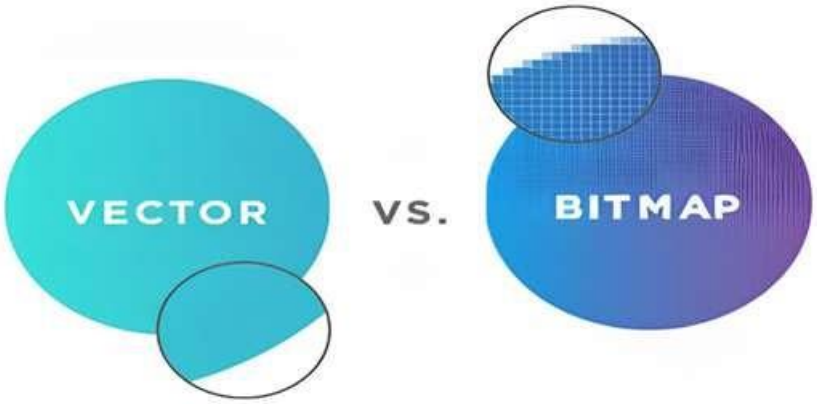

Figure 2: Vector image Vs Bitmap image.

2.1.1. Raster Image

A raster graphic or bitmap image is made up of a matrix of pixels (picture elements), each containing color or grayscale information. These pixels are arranged in a grid pattern, and collectively they form the entire image. The position and value of each pixel determine the image's resolution and visual detail [13].

The resolution of a bitmap image refers to the number of pixels per unit length (typically pixels per inch, PPI). Higher resolutions result in more detailed images, but also increase file size. Importantly, bitmap images are resolution-dependent: enlarging a bitmap without interpolation leads to pixelation and quality loss.

Common raster image formats include:

- **PNG (Portable Network Graphics):** Supports lossless compression and transparency. Ideal for web graphics and interface design but not suited for animation.

- **BMP (Bitmap):** A Windows-native format with minimal compression. Large file sizes make it less practical today.
- **GIF (Graphics Interchange Format):** Limited to 256 colors, supports simple animations. Utilizes compression methods like CLUT and LZW.

2.1.2. Vector Image

Vector images, in contrast, are composed of geometric primitives such as lines, curves, and shapes defined by mathematical formulas. One of the most important vector components is the Bézier curve, which uses control points and handles to define smooth, scalable paths.

Each object in a vector graphic is independent and resolution-independent, allowing for infinite scalability without quality degradation. Vector graphics are ideal for logos, icons, and illustrations but are not typically used for photographs or detailed imagery. [13]

While vector and raster data may coexist (e.g., in layered design files), image stitching focuses exclusively on raster images due to their pixel-based nature.

2.2. Moving Image

A moving image, or digital video, is essentially a sequence of still images (frames) displayed in rapid succession to create the illusion of motion. Each frame is a raster image, typically captured at a constant frame rate—measured in frames per second (FPS) [13].

Two primary methods exist for capturing video frames:

- **Interlaced Scan:** Captures alternate lines (odd then even) in two passes, allowing faster motion sampling but introducing potential artifacts.
- **Progressive Scan:** Captures all lines of the frame simultaneously, resulting in clearer images especially for fast motion.

Digital videos can be copied without quality loss and are typically compressed using various encoding formats. Common codecs include:

- **MPEG-2:** Used for DVDs, offering high quality with moderate compression.
- **MPEG-4 (Part 10 / AVC / H.264):** Widely used for web streaming and HD video.
- **Windows Media Video (WMV):** Proprietary codec optimized for Microsoft environments.

In video stitching, these frames serve as the temporal equivalents of still images. However, stitching them introduces new challenges, such as synchronizing multiple video

sources, managing motion, and ensuring temporal consistency. These aspects will be addressed in the dedicated section on video stitching.

3. Image Stitching: Definition and Process

With the increasing demand for wide-angle and immersive visual content, traditional cameras often fall short due to their limited field of view. To overcome this constraint, image stitching techniques have been developed. These techniques enable the combination of multiple overlapping images into a single, larger visual representation, offering an extended perspective and higher resolution [5].

3.1. Definition of image stitching

Image stitching involves aligning and combining multiple images with overlapping content into a seamless panoramic composite. Typically, this process includes feature detection, geometric transformation estimation, image warping, and blending. The objective is both geometric alignment and photometric coherence, ensuring minimal visible seams or artifacts [3].

Figure 3: Example of image stitching showing the merging of overlapping photographs into a panoramic view.[38]

Image stitching originated in the photographic community, where more manually intensive methods based on surveyed ground control points or manually registered tie points have long been used to register aerial photos into large-scale photo-mosaics. One of the key advances in this community was the development of bundle adjustment algorithms, which

could simultaneously solve for the locations of all of the camera positions, thus yielding globally consistent solutions. Another recurring problem in creating photo-mosaics is the elimination of visible seams, for which a variety of techniques have been developed over the years. In film photography, special camecras were developed in the 1990s to take ultra-wide angle panoramas, often by exposing the film through a vertical slit as the camera rotated on its axis.

In the middle of 1990s, image alignment techniques started being applied to the construction of wide-angle seamless panoramas from regular hand-held cameras. More recent work in this area has addressed the need to compute globally consistent alignments to remove ghosting due to parallax error and object movement, and to deal with varying exposures. These techniques have spawned a large number of commercial stitching products.[3]

In the next sections, we explore the classification of stitching methods and their respective processing pipelines.

4. Different kinds of Image Stitching

Image stitching can be categorized into several types based on how the images are captured and the intended structure of the final panorama. The main kinds include:

4.1. Mosaic

Stitch multiple rows of pictures that were taken without rotating the camera around a single point, but with the camera kept perpendicular with the subject.[12]

Figure 4: Mosaic stitching technique combining multiple overlapping images without camera rotation to produce a wide field of view composite.[37]

4.2. Panorama (singlerow)

Stitch a single row of pictures (created by rotating the camera around a single point in a flat plane, which is normally parallel with the horizon).

4.3. Panorama (multi row)

Stitch multiple rows of pictures (created by rotating the camera around a single point in a flat plane but tilting or pitching the camera up and/or down so that for each row of pictures the lens is not necessarily parallel with the plane of rotation).

4.4. Panorama (panocamera)

Just stitch together the ends of panoramic picture created with a panoramic camera.

4.5. Spherical panorama

Stitch any number of pictures in such a way as to create a spherical panorama, the important distinction between this and single or multi row panoramas is that the "poles" (i.e. the very top and bottom of the image) must be stitched also so that the user can look straight up and down and see a smoothly blended image.

5. Classification of Image Stitching

Image stitching has evolved into a multidisciplinary technique applied in domains ranging from panoramic photography and medical imaging to satellite mapping and virtual reality. Over the years, a variety of approaches have been developed to address different technical constraints, data types, and application requirements. Accordingly, image stitching methods can be classified along several axes, including [4]:

- The type of features used (pixels vs. keypoints),
- The nature of camera motion (static vs. dynamic),
- The projection model employed (planar, cylindrical, spherical),
- The computational domain (spatial vs. frequency),
- And the intended application context (offline mosaics, real-time video, etc.).

Among these, one of the most widely accepted and technically significant classifications is based on the type of registration algorithm used to align the images. Registration is a critical step in the stitching pipeline, responsible for determining how input images correspond to each other based on shared visual content.

From this perspective, image stitching techniques are broadly divided into two categories:

- Spatial domain-based methods, which operate directly on image data in the spatial (pixel) domain,
- Frequency domain-based methods, which leverage the mathematical properties of image transforms (typically the Fourier transform) to perform alignment.

Within the spatial domain, two subcategories dominate: area-based methods, which rely on comparing raw pixel intensities, and feature-based methods, which extract and match salient keypoints or descriptors across images. These approaches differ in their assumptions, computational complexity, and robustness to scale, rotation, or illumination changes.

The following sections present these categories in greater detail, highlighting their principles, strengths, and limitations.

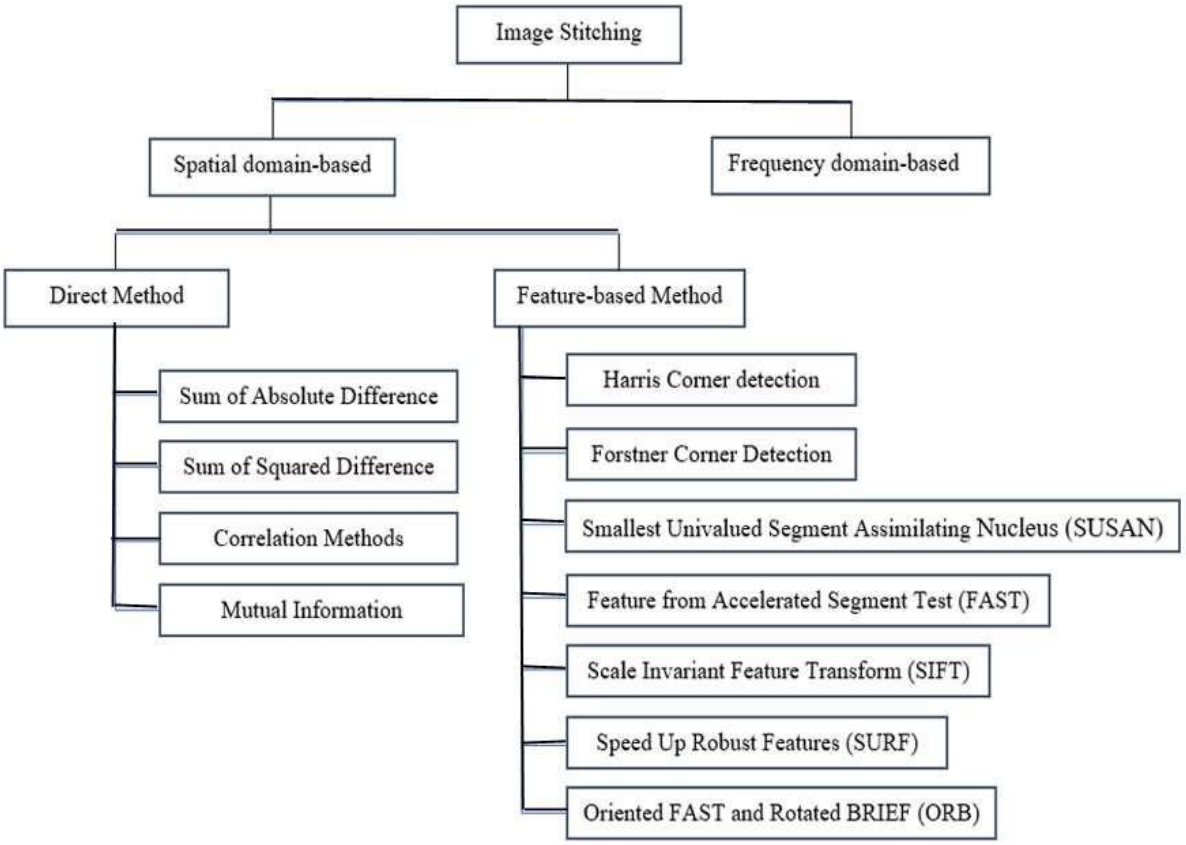

Figure 5: Classification of image stitching techniques [4]

5.1. Spatial Domain-based Image Stitching

This category of image stitching uses pixel intensity properties to perform image registration. In the spatial domain, image stitching may be area-based or feature-based. Pixel-based or direct image stitching is another name for area-based stitching/mosaicking. The

matching operation is carried out by comparing each pixel between two images rather than the feature-to-feature matching [4].

5.1.1. Direct Method

To find the overlapping region between the two images, the following process is used: Direct method/area-based image stitching method compares all the intensities of all the pixels of the two images. Variety of similarity measuring techniques like Sum of squared difference, Sum of absolute difference, Correlation method, Mutual information technique etc can be used to derive the matching region.

5.1.1.1. Sum of Absolute Differences (SAD)

The Sum of Absolute Differences (SAD) is a basic yet effective method commonly used in motion estimation. It calculates the total absolute intensity difference between two images I0 and I1 as follows:[14]

$$\mathrm{SAD}(u) = \sum_i |I_0(x_i + u) - I_1(x_i)|$$

SAD provides a straightforward way to detect translational displacement between similar images but is sensitive to noise and intensity variations.

5.1.1.2. Sum of Squared Differences (SSD)

The Sum of Squared Differences (SSD) is another fundamental method for determining overlap by minimizing the squared difference between pixel values in two images:[15]

$$\mathrm{SSD}(u) = \sum_i [I_2(x_i + u) - I_1(x_i)]^2$$

While SSD is easy to implement and computationally efficient, it is highly sensitive to illumination changes.

5.1.1.3. Correlation Methods

Cross-correlation measures similarity between image patches and is often used for block-based matching [4]:

$$C(m,n) = \sum_x \sum_y I_0(x,y) \cdot I_1(x-m, y-n)$$

However, basic correlation is affected by brightness differences. To overcome this, Normalized Cross-Correlation (NCC) is used, which normalizes pixel intensities to provide a more stable comparison:

$$\mathrm{NCC}(u) = \frac{\sum_i [I_0(x_i) - \bar{I}_0] \cdot [I_1(x_i + u) - \bar{I}_1]}{\sqrt{\sum_i [I_0(x_i) - \bar{I}_0]^2 \cdot \sum_i [I_1(x_i + u) - \bar{I}_1]^2}}$$

NCC values range from -1 to 1 and are more robust to changes in illumination, but this method still lacks invariance to scale and rotation.

5.1.1.4. Mutual Information (MI)

Mutual Information (MI) measures the statistical dependency between two images. It is often used in multi-modal image registration, where intensity values do not match directly:

$$MI(I_1, I_2) = H(I_2) - H(I_2|I_1)$$

Here, H denotes entropy, and the conditional entropy depends on the joint probability distribution of the pixel intensities. While MI performs well in heterogeneous data scenarios, its effectiveness drops with low-resolution images or small overlapping areas. Moreover, MI-based approaches tend to have a narrow convergence range.

5.1.2. Feature-Based Method

In this method, feature points in the images, such as corners, blobs, are extracted, and these extracted feature points descriptors are compared with each other to find the overlap area. The correspondences between images and homography can be measured from the locations of the extracted matching feature points. Images are first warped and then aligned into a final frame using homography matrices [7].

Among the spatial domain approaches, feature-based methods are widely used and well-documented, making them a practical choice for illustrating the stitching pipeline. The following section presents the typical steps involved in a feature-based image stitching process, as a representative example of spatial domain methods.

5.1.2.1. Steps in Feature-Based Method

The fundamental steps in feature-based image stitching include image acquisition, feature detection and matching, homography estimation, image warping, and image blending. Figure 6 illustrates the various phases involved in this process [16].

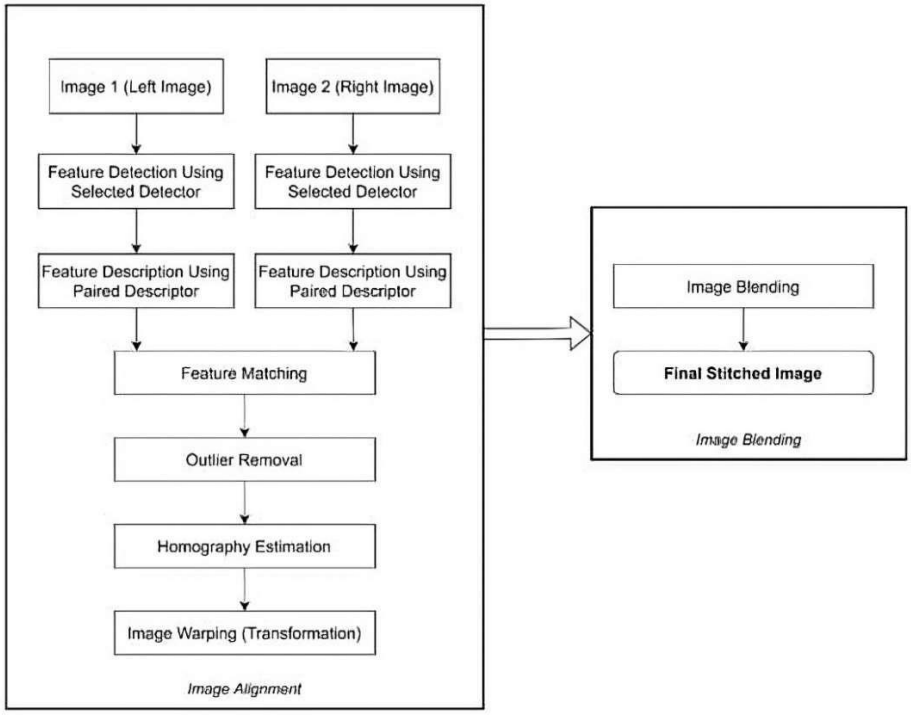

Figure 6: Step by step workflow of the feature-based image stitching process [11].

a. Image Acquisition

For any stitching application, multiple input images are fed to the system. Input images are collected by any acquisition method. The first step in the image stitching process is image acquisition. It is the method of collecting an image from a variety of sources. Images for panoramic photography can be captured by moving a camera from sequential directions. The movement of the camera should be parallel to the scene. Another way of capturing images is rotating a camera around its vertical axis or using a handheld camera [9].

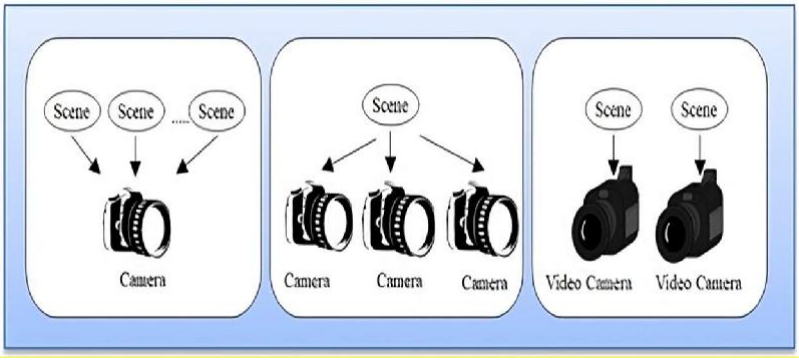

Figure 7: Different camera positioning techniques for acquiring overlapping images used in stitching.[19]

b. Feature Detection

A feature is a piece of information which is relevant for solving the computational task related to a certain application. Features may be specific structures in the image such as points, edges or objects [11]. Features may also be the result of a general neighborhood operation or feature detection applied to the image. The features can be classified into two main categories:

- The features that are in specific locations of the images, such as mountain peaks, building corners, doorways, or interestingly shaped patches of snow. These kinds of localized features are often called keypoint features (or even corners) and are often described by the appearance of patches of pixels surrounding the point location.

  - The features that can be matched based on their orientation and local appearance (edge profiles) are called edges and they can also be good indicators of object boundaries and occlusion events in the image sequence.

Figure 8: An image (left) and its corresponding features detected (right)

c. Feature descriptor

A feature descriptor is an algorithm which takes an image and outputs feature descriptors/feature vectors. Feature descriptors encode interesting information into a series of numbers and act as a sort of numerical "fingerprint" that can be used to differentiate one feature from another [2].

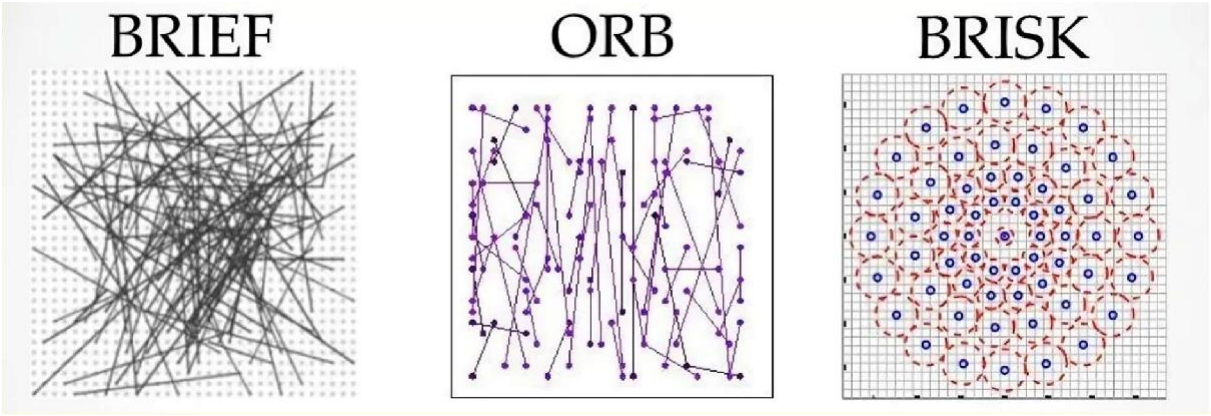

Figure 9: Visual Comparison of Binary Feature Descriptors: BRIEF, ORB, and BRISK.

Ideally, this information would be invariant under image transformation, so we can find the feature again even if the image is transformed in some way. After detecting interest points we go on to compute a descriptor for every one of them. Descriptors can be categorized into two classes:

- Descriptors try to resemble shape and appearance only in a local neighborhood around a point and thus are very suitable for representing it in terms of matching.

- Global Descriptor: A global descriptor describes the whole image. They are generally not very robust as a change in part of the image may cause it to fail as it will affect the resulting descriptor.

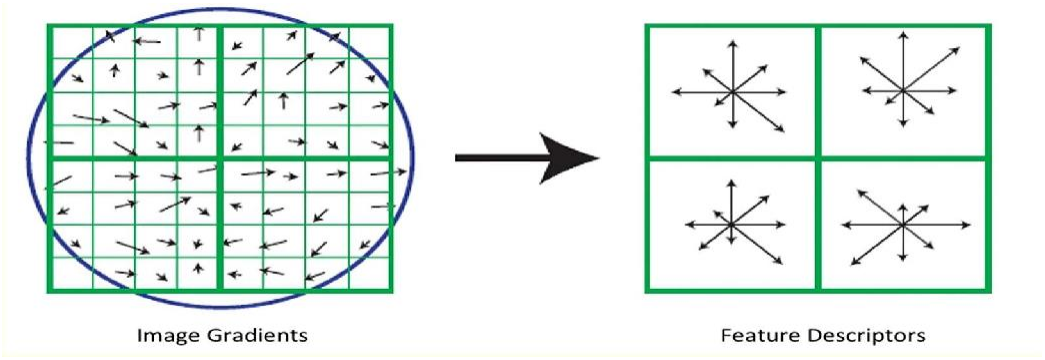

Figure 10: Schematic representation of the computation of a local feature descriptor from a keypoint region in an image.[13]

d. Matching features

Features matching or generally image matching, a part of many computer vision applications such as image registration, camera calibration and object recognition, is the task of establishing correspondences between two images of the same scene/object. A common approach to image matching consists of detecting a set of interest points each associated with image descriptors from image data. Once the features and their descriptors have been extracted from two or more images, the next step is to establish some preliminary feature matches between these images [7].

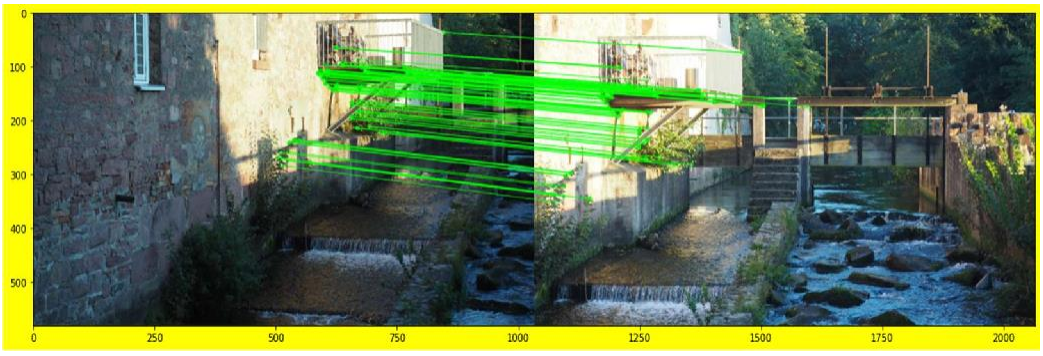

Figure 11: Feature matching between two images for image stitching.

Generally, the performance of matching methods based on interest points depends on both the properties of the underlying interest points and the choice of associated image descriptors. Thus, detectors and descriptors appropriate for images

contents shall be used in applications. For instance, if an image contains bacteria cells, the blob detector should be used rather than the corner detector. But, if the image is an aerial view of a city, the corner detector is suitable to find man-made structures. Furthermore, selecting a detector and a descriptor that addresses the image degradation is very important.

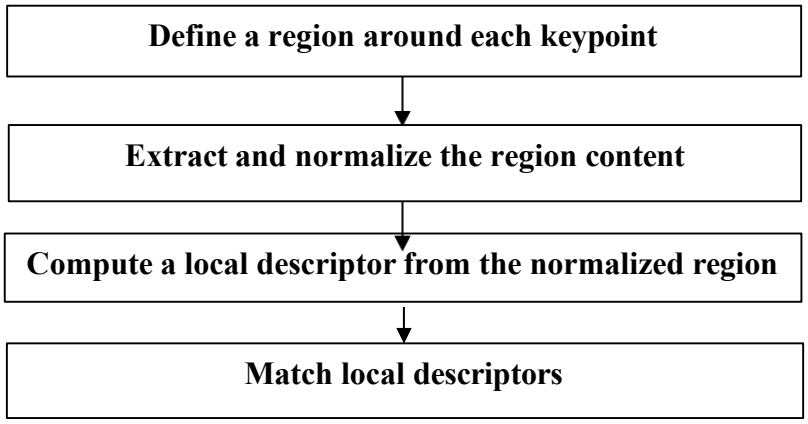

Figure 12: Keypoint Detection and Descriptor Matching Workflow.

Having outlined the fundamental steps involved in the feature-based stitching method image acquisition, feature detection, feature description, and feature matching we will now delve deeper into the core of the stitching process: feature detection and description algorithms. Several robust and widely-used algorithms have been developed to detect distinctive keypoints and compute reliable descriptors.

- **Scale Invariant Feature Transform (SIFT)**

The Scale-Invariant Feature Transform (SIFT) is a widely used technique in computer vision for detecting and describing local features in images. It was introduced by David Lowe in 1999 and has since become a fundamental tool for various applications, such as object recognition, image stitching, and 3D reconstruction [16].

RANSAC algorithm is used for outlier detection and to compute the transformation parameter [1]. Images are warped and aligned to create the final composite image using the transformation parameter obtained. The SIFT algorithm is good for stitching high-resolution images with rotation, Scale, and affine motion changes. However, one drawback is the lengthy processing time.

- **Feature from Accelerated Segment Test (FAST)**

Rosten E. and Drummond T. introduced FAST, a fast corner and interest point detection algorithm, in 2006 [19]. They used a Machine Learning approach to hasten the corner detection process [2]. The FAST detector compares pixels on a circle with a fixed radius centred on a point to detect interest points. The FAST algorithm considers a 16-pixel circle around corner candidate p, as shown in figure 13.

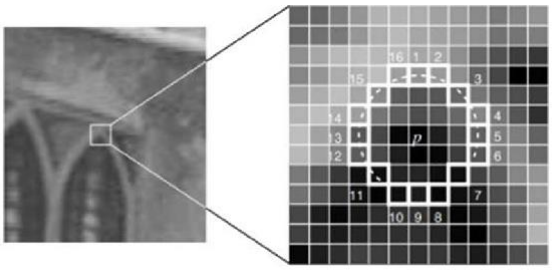

Figure 13: The corner detection by using the FAST algorithm [2]

- **Speeded Up Robust Features (SURF)**

Herbert Bay et al. introduced SURF in 2006. SURF is a faster algorithm than SIFT, and it is said to be more resistant to image transformations. The first step in the SURF algorithm is to select key points such as blobs, corners and T-junctions in the image. A distinctive feature vector is used to represent the neighbourhood of the key point. The descriptor should be distinct and resistant to noise, detection errors, and geometric distortions. Key points are compared by matching the feature vectors [17].

The SURF algorithm uses integral images and the Sum of 2D Haar wavelet responses. It employs an integer approximation to the Hessian blob detector's determinant, which is determined using an integral image of the source image. Fast computation, suitability for real-time monitoring, and object recognition are the key advantages of the SURF algorithm. It accelerates the SIFT detection process by ensuring that the detected key points are of high quality. The processing speed of feature vector matching is increased. The Hessian matrix is used in conjunction with descriptors to minimize the dimensionality of the descriptors, resulting in a faster matching process. The limitation of SURF is that poor at handling viewpoints and illumination changes.

- **Oriented FAST and Rotated BRIEF (ORB)**

The ORB algorithm is based on the Binary Robust Independent Elementary Features (BRIEF), which is an extremely fast key-point descriptor. The binary-based features are better than vector-based features in terms of computation speed, storage, comparison efficiency. The ORB descriptor uses the well-known FAST key-point detector as its foundation. These methods are effective due to their high efficiency and low cost. ORB provides a fast and accurate orientation component [8].

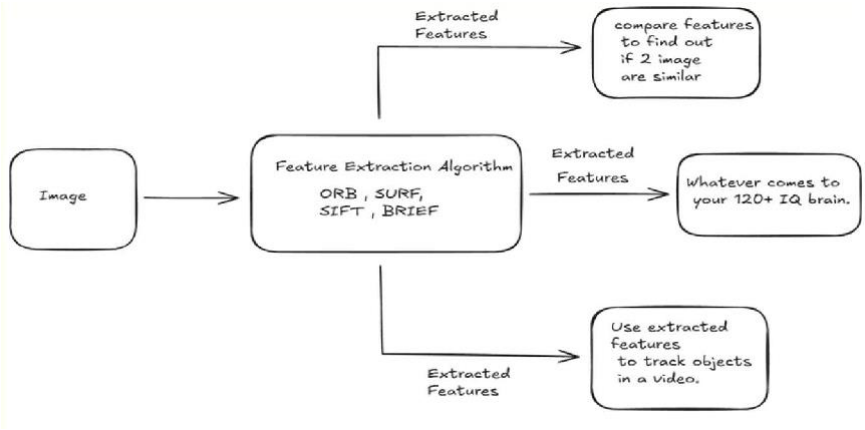

Figure 14: Applications of Feature Extraction Algorithms in Computer Vision.

In conclusion, algorithms like SIFT, ORB, FAST, and SURF provide powerful tools for reliable keypoint detection and matching, each offering unique strengths tailored to different stitching scenarios. Choosing the right algorithm depends on the specific requirements, such as accuracy, computational speed, and robustness to image variations.

Table 2: Comparison of Feature Detection Algorithms for Image Stitching.[42]

Method	Speed	Accuracy	Invariance	License	Use in Stitching
SIFT	Slow	Very High	Scale + Rotation	Free (expired)	Best for precision
ORB	Fast	Moderate	Rotation only	Free (OpenCV)	Great for real-time use
FAST	Very Fast	Low	None	Free (OpenCV)	Detector only (needs BRIEF)
AKAZE	Fast	High	Scale+rotation	Free(OpenCV)	Excellent speed/accuracy trade-off

e. Homography estimation using RANSAC

Homography is a transformation between two image planes. It maps the points of an image in one plane to the corresponding points in the image of another plane.[18] For an image point P, the 3x3 transformation matrix H transforms this point P to P' using P' = HP where H is represented in the homogeneous coordinates system as follows:

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

Let us assume the point PPP in homogeneous coordinates is:

$$P = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Then, the corresponding transformed point P′ is given by:

$P' = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$ in homogeneous coordinates. Then we can write :

$$c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

By eliminating c, we can formulate the above equation in the form Ah = 0, where,[1]

$$A = \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & ux & uy & u \\ 0 & 0 & 0 & -x & -y & -1 & vx & vy & v \end{bmatrix}$$
$$h = \begin{bmatrix} h1 & h2 & h3 & h4 & h5 & h6 & h7 & h8 & h9 \end{bmatrix}^T$$

Knowing A, we can find the values of h by solving the above equation. It is to be noted that we need at least four such point correspondences in order to estimate the homography matrix. In real-world image stitching applications, feature matching between two images often includes incorrect correspondences known as outliers.

RANSAC works by iteratively selecting random subsets of feature point correspondences (typically four points) and estimating a candidate homography matrix H. For each iteration, the algorithm counts how many points from the entire dataset fit the estimated model within a defined error threshold (typically between 1.0 and 10.0 pixels, depending on image resolution). The model with the maximum number of inliers is retained. Finally, a refined homography is computed using least-squares optimization on all inliers.

Although RANSAC is widely used due to its robustness to high outlier ratios, it is a non-deterministic algorithm, meaning it does not guarantee the optimal solution in every run. For higher accuracy, proper feature correspondence filtering is critical, and RANSAC is often complemented by other methods such as Least Median of Squares.

Figure 15: Key point and inlier matching using RANSAC.[19]

f. Image Warping

Once the homography matrix H has been estimated, the next critical step in the feature based stitching pipeline is image warping. This operation geometrically transforms one image so that it aligns properly with another image in the mosaic. It plays a key role in ensuring that corresponding regions between overlapping images are projected onto a common coordinate system [3].

- **Geometric Transformation**

The warping transformation is based on the homography:

$$\mathbf{x}' = \mathbf{H} \cdot \mathbf{x}$$

Where :

- $\mathbf{x} = [x, y, 1]^t$ is a point in the source image (in homogeneous coordinates)
- $\mathbf{x}' = [u', v', w]^t$ is the transformed point in the destination image
- $\mathbf{H}$ is a 3×3 projective transformation matrix

The actual coordinates in the destination image are obtained by normalization:

$$u = u' / w, \quad v = v' / w$$

- **Forward Warping (Alternative View)**

Alternatively, in **forward warping**, each pixel $\mathbf{x}$ in the source image is directly transformed using:

$$\mathbf{x}' = \mathbf{H} \cdot \mathbf{x}$$

However, this approach often leads to gaps or holes in the output image, since not all destination pixels are reached. To mitigate this issue, techniques like splatting are used, where contributions are distributed to surrounding pixels and normalized [1].

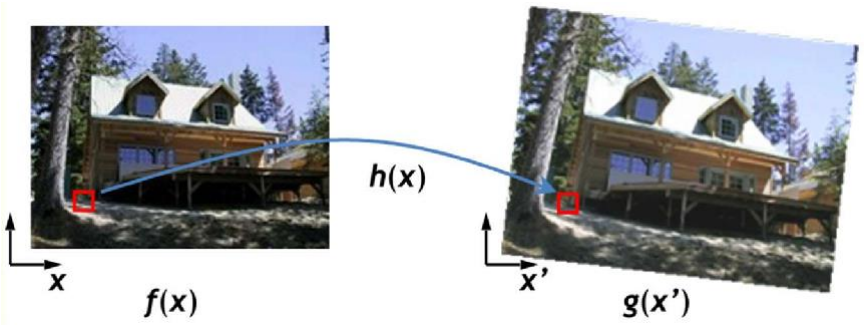

Figure 16: Forward Warping in Image Transformation Using Homography.[38]

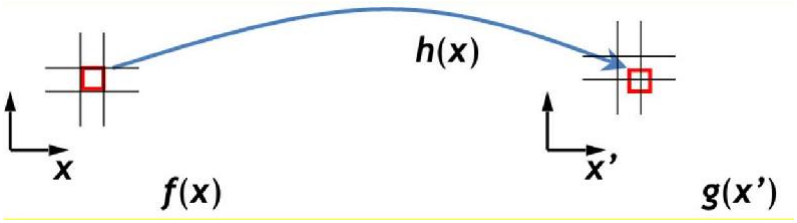

Figure 17: Forward Mapping in Image Warping Using Homography Transformation.[38]

- **Inverse Warping (Pixel Mapping)**

In practice, we use inverse warping to avoid holes in the output image. For every pixel $\mathbf{x}'$ in the destination image [1], we compute its corresponding location $\mathbf{x}$ in the source image using the inverse homography:

$$\mathbf{x} = \mathbf{H}^{-1} \cdot \mathbf{x}'$$

This ensures that all pixels in the destination image are filled with valid values.

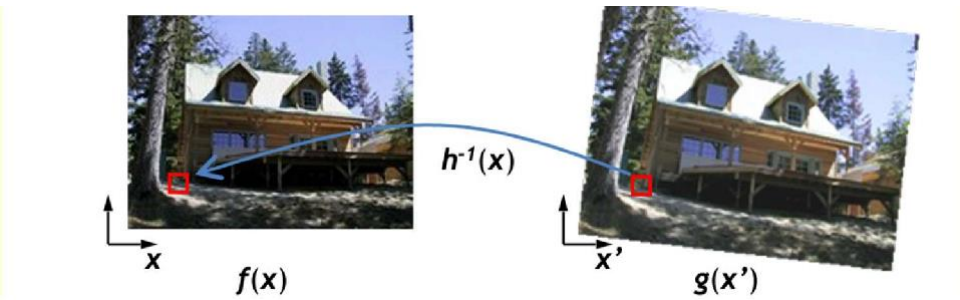

Figure 18: Homography Mapping Between Two Images for Geometric Alignment.[38]

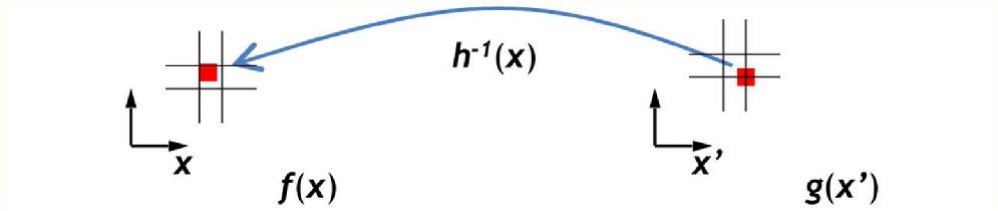

Figure 19: Inverse Warping in Image Transformation Using Homography.[38]

- **Resampling and Interpolation**

Since $\mathbf{x}$ typically maps to non-integer coordinates, the color value at that location must be estimated using interpolation. Common interpolation methods include:

- **Nearest-neighbor interpolation** (fast but lower quality)
- **Bilinear interpolation** (smooth and commonly used)

- **Bicubic interpolation** (higher quality, more computational cost)

g. Blending Image

Blending is another key step of image stitching. It is used to remove any apparent seams that may appear in the final composite image due to misalignments, camera exposure differences, scene lighting variations, or the presence of moving objects between frames and lack of proper geometric alignment [16]. If registration is done perfectly and there are no exposure differences in input images, blending is an easy task. There are different types of blending techniques. Transition smoothing and optimum seam finding are the two types of blending techniques. Feathering and alpha blending are other terms for transition smoothing. Feathering, pyramid blending, and gradient based blending are common blending methods that use transition smoothing.

h.Post-processing

After blending in image stitching, the next crucial step is post-processing to refine the final panorama and ensure visual consistency. This includes exposure compensation to correct any brightness or color mismatches between overlapping images, and seam optimization to further minimize visible transitions, especially in dynamic or poorly aligned regions. If geometric distortions remain, geometric correction or warping refinement is applied to straighten the horizon or correct curved structures. Finally, cropping or content-aware filling is used to remove black borders or fill missing regions, producing a clean, seamless panoramic output ready for visualization or export.

5.2. Frequency Domain-Based Image Stitching

Frequency domain image stitching techniques are primarily based on the Fourier Transform (FT). The FT of two images can be correlated by multiplying the Fourier coefficients of one image with the complex conjugate of the other. Phase correlation, a prominent frequency domain registration method, utilizes this principle to identify overlapping regions between input images.

Specifically, the overlapping area is determined by computing the element-wise product of the Fourier transform of one image with the complex conjugate of the Fourier transform of the second image. Traditional cross-correlation-based image stitching methods typically handle images displaced either horizontally or vertically to find the overlap.

However, this approach is limited when dealing with displacements in both directions simultaneously.

Phase correlation methods address this limitation by enabling the registration of images with both horizontal and vertical shifts at the same time. Numerous image registration algorithms leveraging phase correlation have been proposed in the literature. Additionally, Fourier-based techniques are valued for their computational efficiency, often accelerating the process of image registration [16].

6. Applications of Image Stitching

Image stitching has evolved into a powerful and practical technique across a wide range of domains, enabling the creation of seamless, wide-field visual representations that extend beyond the field of view of individual cameras. Its ability to combine multiple overlapping images into a single, coherent composite makes it essential in both scientific and commercial applications.

6.1. Panoramic Photography

One of the most widespread applications of image stitching is in the creation of panoramic photographs. By stitching together images taken from a fixed viewpoint while rotating the camera, high-resolution, wide-angle landscapes can be produced. This technique is now embedded in many modern smartphone cameras and consumer software applications [26].

Figure 20: Panorama stitching.

6.2. Virtual Tours and Street View

Image stitching is integral to services like Google Street View, virtual tourism, and real estate walkthroughs. These applications generate immersive 360° scenes by stitching images captured from multiple angles, allowing users to navigate environments as if they were physically present.

6.3. Aerial and Satellite Imaging

In remote sensing and geospatial analysis, multiple aerial or satellite images are stitched to create large-scale, high-resolution maps. This is critical for agriculture, urban planning, disaster monitoring, and environmental observation, where wide and continuous coverage is essential.

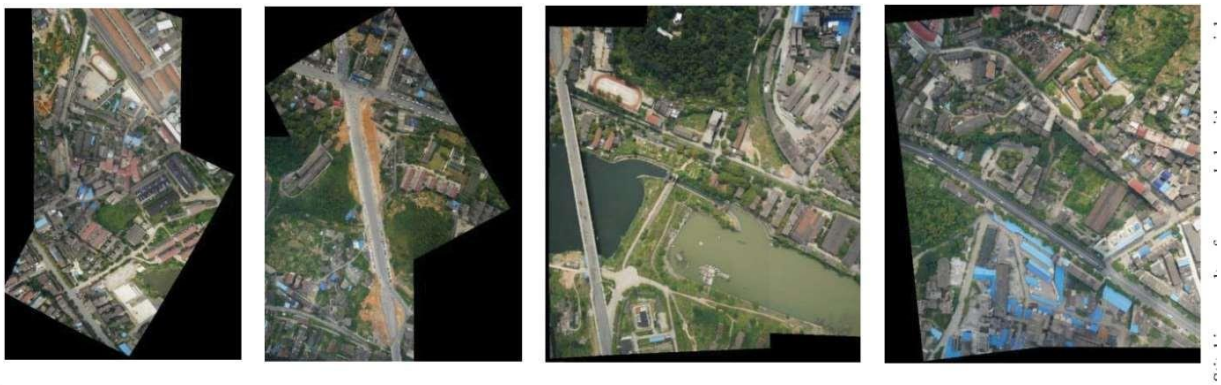

Figure 21: Stitching results of proposed algorithm on aerial sample images.[39]

6.4. Medical Imaging

Stitching techniques are employed in medical diagnostics to reconstruct complete views of anatomical regions from several microscopic or radiographic image slices. This is particularly useful in pathology, ophthalmology (e.g., retinal mosaics), and dermatology.

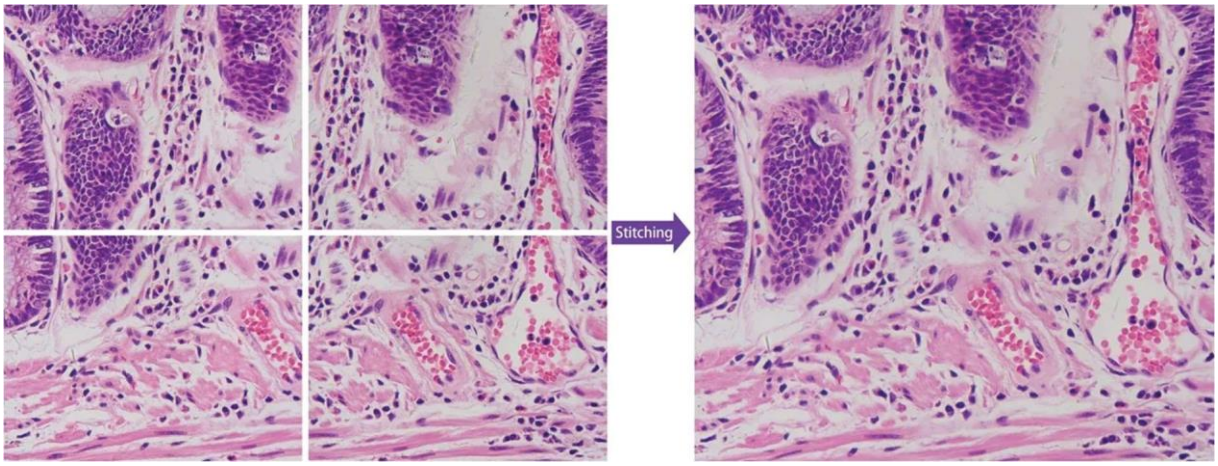

Figure 22: Application of image stitching in medical imaging to reconstruct a complete anatomical view from multiple microscope images.[40]

6.5. Cultural Heritage and Document Digitization

Museums, libraries, and research institutions use stitching to digitize large artworks, manuscripts, and historical documents. Multiple high-resolution images are combined to preserve and analyze artifacts in their entirety without physical handling.

6.6. Scientific Visualization and Microscale Imaging

In fields like biology and material science, microscopes capture tiled images of small samples. Stitching these into a single image allows researchers to analyze large regions at microscopic resolution without losing contextual information.

The wide range of applications discussed above illustrates the maturity and effectiveness of image stitching in various fields. However, as visual systems become more dynamic and interactive especially in areas like real-time surveillance, virtual reality, and autonomous navigation static image stitching is no longer sufficient. These emerging demands have led to the evolution of video stitching, which builds on image stitching principles while addressing time-dependent challenges.

7. From Image Stitching to Video Stitching

While image stitching focuses on combining static images into a seamless panorama, video stitching extends these principles to moving visual sequences captured by multiple cameras. This temporal dimension introduces new complexities, such as frame synchronization, motion handling, and temporal consistency. Unlike static images, video sequences may include dynamic scenes, object motion, and varying lighting conditions that make alignment and blending more challenging [14].

Video stitching aims to produce coherent panoramic or 360° videos by aligning and blending overlapping video streams in real time or near-real time. Applications range from immersive virtual reality and autonomous vehicles to surveillance and live broadcasting.

Due to these additional challenges, video stitching has become a distinct and active field of research, building upon but going beyond the traditional image stitching techniques.

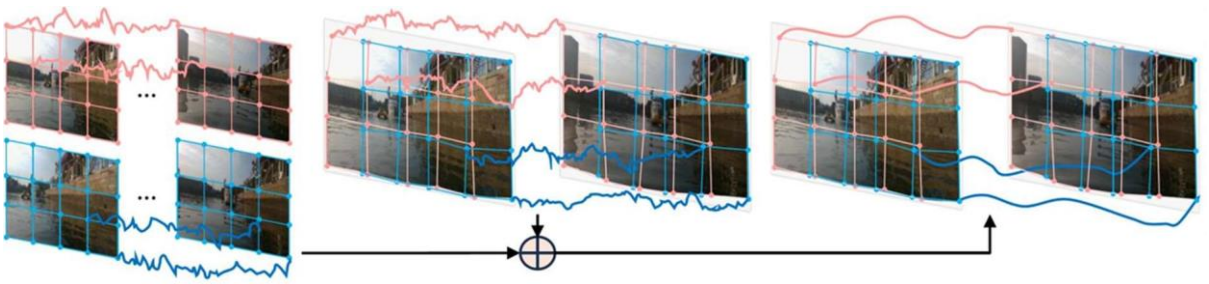

Figure 23: Video stitching.

Conclusion

In summary, this chapter has laid the groundwork for understanding image stitching by exploring its historical evolution, fundamental principles, core algorithms, and diverse applications. We examined the structure of digital images and videos, detailed the complete stitching pipeline from feature detection and matching, to homography estimation, warping, blending, and post-processing and reviewed widely used techniques like SIFT, SURF, ORB, and FAST. Additionally, we classified image stitching methods into spatial and frequency domain approaches, highlighting their strengths and limitations in real-world scenarios. However, while image stitching focuses on static visual content, modern applications increasingly require stitching across temporal sequences, where additional challenges arise due to motion, synchronization, and real-time constraints. These complexities mark the transition from image to video stitching, a field that builds upon but significantly extends traditional techniques. The next chapter provides a state-of-the-art review of video stitching approaches both conventional and deep learning-based analyzing how they tackle critical challenges such as temporal alignment, parallax handling, and dynamic scene processing in real time.

Chapter II

Video Stitching

Introduction

Building on the principles discussed in the previous chapter, this chapter explores the current state of video stitching techniques. It outlines definition and scope of video stitching, the key concepts and challenges unique to video data, highlights major application areas, and reviews traditional and deep learning-based approaches. A comparative analysis of recent methods is presented to identify the most effective strategies for achieving accurate, stable, and real-time panoramic video stitching.

1. Definition and scope of video stitching

Video stitching is the process of combining multiple synchronized video streams typically captured by cameras with partially overlapping fields of view into a single, continuous panoramic or 360-degree video sequence. Unlike image stitching, which operates on static images and focuses purely on spatial alignment, video stitching introduces a temporal dimension, requiring the preservation of consistency across consecutive frames while accommodating motion and dynamic content. This technique is widely employed in domains that demand extended or immersive fields of view, such as virtual reality, automotive surround-view systems, surveillance, and aerial mapping.

What distinguishes video stitching from its image-based counterpart is the presence of additional challenges related to motion, time, and scale. These include ensuring temporal coherence between frames, managing moving objects that may cross stitching seams, and meeting the demands of real-time processing in critical systems.

Due to this increased complexity, video stitching has become a distinct and evolving research field that combines principles from geometry, signal processing, computer vision, and machine learning. The remainder of this section introduces the key technical concepts that underpin video stitching pipelines.

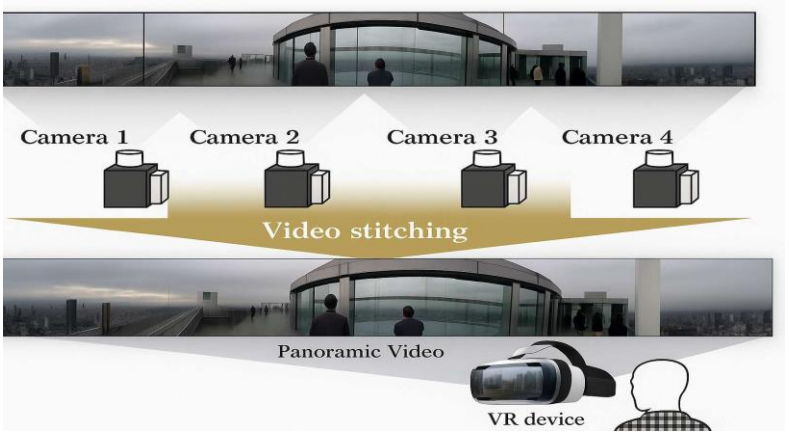

Figure 24: Multi-Camera video stitching for immersive panoramic output.

1.1. Key concepts

In order to fully grasp the intricacies of video stitching, it is essential to understand the core concepts that underpin the technology. These concepts not only define how video frames are aligned and stitched together but also address the challenges posed by dynamic scenes, camera motion, and real-time processing. Below are the key concepts that are foundational to the video stitching process, which will be explored in more detail throughout this chapter.

1.1.1. Geometric alignment

- **Homography**

A homography is a 3×3 projective transformation that maps points in one image plane to another, assuming a planar scene or pure camera rotation. Denoted $x'=Hx$, it has eight degrees of freedom and is estimated via feature correspondences and algorithms like RANSAC. Homographies serve as the backbone for aligning frames in the video stitching pipeline.

- **Parallax**

Parallax occurs when objects at different depths move inconsistently across different viewpoints it is a primary source of misalignment in stitching . Because standard homography assumes planarity, scenes with depth variation exhibit ghosting or visible seams. Mitigating parallax often requires advanced techniques like mesh-based warping or seam- cutting algorithms.

- **Field of view (FoV)**

The angular extent of the observable scene captured by a camera. In stitching, overlapping FoVs are essential to enable registration and seamless merging. Wide-FoV or fisheye lenses are often used in surround-view setups.

- **Camera calibration**

The process of estimating intrinsic and extrinsic parameters of a camera to correct distortion and align frames properly. Accurate calibration improves stitching quality, especially for wide-FoV lenses .

1.1.2. Temporal consistency

- **Temporal coherence**

A stitched video must preserve smooth transition between successive frames. Temporal coherence ensures that both alignment and blending remain consistent over time—otherwise viewers perceive flicker, jitter, or instability .Achieving this often requires synchronization and smoothing of transformations across frames.

- **Jitter**

Jitter is a temporal misalignment artifact where the visual content appears to shake, bounce, or shift inconsistently between frames. It is often caused by unstable warping transformations or inconsistent homographies applied over time. In stitched videos, jitter disrupts visual continuity and undermines the perceived stability of the scene, especially in panoramic or 360° footage.

- **Seam drift**

A temporal artifact where the optimal stitching seam shifts between frames, causing visual discontinuity or tearing. Requires dynamic seam refinement or motion-aware warping [24].

- **Optical flow**

A technique for estimating pixel-wise motion between consecutive frames. Used for temporal smoothing, dynamic scene understanding, and motion-aware blending in video stitching [25].

1.1.3. Motion and dynamic scene handling

- **Motion Handling**

Video inputs frequently contain dynamic scenes moving people, vehicles, or objects crossing camera boundaries. Avoiding motion artifacts (e.g., ghosting, duplication) demands either motion-aware blending or dynamic masking. This contrasts with image stitching, where motion is rarely a concern.

- **Ghosting**

Ghosting is a spatial artifact that appears when moving objects are misaligned between overlapping views, causing duplicate or semi-transparent "ghost-like" appearances in the stitched output. It usually results from improper handling of parallax, poor synchronization, or naive blending techniques that fail to account for dynamic content. Ghosting is especially noticeable when people or vehicles cross stitching seams in the scene [26].

- **Flicker**

Flicker refers to visible and abrupt changes in brightness, color, or exposure between consecutive video frames. In the context of video stitching, flicker occurs when blending or alignment varies inconsistently over time, leading to an unstable or distracting viewing experience. It typically results from differences in lighting conditions, camera auto-exposure, or poor temporal coherence in the stitching pipeline [27].

1.1.4. Computational aspects

- **Real-Time Constraints**

Many applications (e.g., surveillance, VR, automotive systems) require stitching at real-time or near-real-time speeds. This introduces computational challenges related to resource limits, latency requirements, and hardware acceleration (e.g., GPU, DSP).

- **Parallelism and optimization**

Modern stitching pipelines often exploit multi-threading, SIMD, and GPU parallelism to accelerate feature extraction, matching, and blending stages.

Having explored the definition, scope, and key concepts of video stitching, it's essential to understand the actual process behind creating seamless panoramic videos. This involves several key stages, from extracting features to final rendering, all of which work together to address the complexities introduced by dynamic scenes and real-time processing. The following section outlines the typical video stitching pipeline, providing a detailed look at the steps involved in transforming multiple video streams into a unified output.

2. Video stitching pipeline

The video stitching process is typically structured in a multi-stage pipeline that transforms raw footage from multiple cameras into a unified panoramic video. Understanding the key stages of this process ranging from feature extraction to final rendering helps us appreciate the complexity involved in creating a seamless, immersive video. In this section, we will break down each stage, highlighting the methodologies used at each step to ensure optimal video stitching.

2.1. Feature extraction and matching

2.1.1. Feature detection

Similar to image stitching, distinctive features (corners, edges, blobs) are detected in each frame. Traditional algorithms like SIFT, SURF, or ORB are commonly employed; recent approaches might use deep learning methods for more robust feature extraction under challenging conditions.

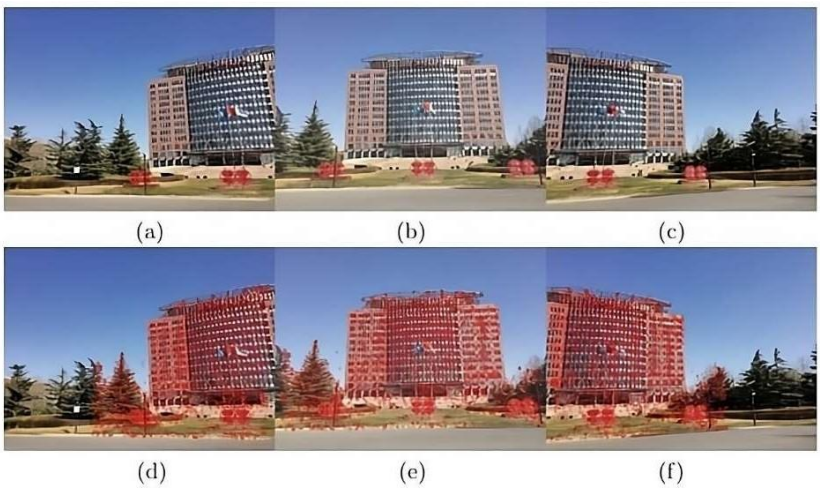

Figure 25: Surf keypoints of multiple video images.[29]

2.1.2. Feature matching and outlier rejection

Once features are extracted, corresponding points in overlapping regions must be identified. Techniques such as RANSAC (Random Sample Consensus) help reject outliers, ensuring that only consistent matches inform the subsequent transformation estimation. This is particularly crucial in dynamic scenes where objects may move or when lighting conditions vary [28].

Figure 26: The schematic plot of matched feature points [29]

2.2. Transformation estimation and frame warping

2.2.1. Computing geometric transformations

With reliable feature correspondences, transformation models (most often homographies for planar approximations) are computed. These models mathematically determine how to map one view onto another. In video stitching, the transformation must remain stable over consecutive frames to prevent noticeable jitter [30].

2.2.2. Frame warping

Using the estimated transformations, each frame is warped to a common coordinate system. This warping aligns the overlapping regions from different video streams. When dealing with video, ensuring smooth transitions from frame to frame is essential to maintain temporal consistency.

2.3. Seam finding and blending

2.3.1. Seam determination

Overlapping frame regions need to be merged in a way that minimizes visible boundaries. Algorithms like graph cuts help determine optimal seams where the differences between the overlapping regions are minimized [31].

2.3.2. Blending techniques

Once seams are found, blending methods (such as multi-band blending or alpha blending) are applied. These techniques smooth out transitions between frames, avoiding abrupt changes in brightness or texture that otherwise might attract the viewer's attention [32].

2.4. Temporal consistency and stabilization

2.4.1. Temporal smoothing

Because videos are sequences of frames, maintaining smooth temporal transitions is vital. Temporal filtering (using methods like Kalman filters or deep-learning–based smoothing) reduces inter-frame jitter. This stabilization step is necessary to address cumulative errors especially pertinent in handheld video footage or in scenes with significant camera movement [22].

2.4.2. Joint optimization strategies

Modern approaches often integrate the stabilization process directly with the stitching. For instance, frameworks combining unsupervised deep learning can concurrently stabilize and stitch video streams. These methods utilize strategies like Markov trajectory smoothing and dynamic attention masks to mitigate errors over time, enhancing visual coherence in dynamic environments [33].

2.5. Real-time processing and hardware optimization

2.5.1. Algorithmic optimizations and parallelism

In applications like 360-degree live streaming, autonomous driving, or surveillance, video stitching must be performed in real time. To achieve this, video stitching pipelines

incorporate algorithm-level optimizations limiting computationally expensive operations and may leverage digital signal processors (DSPs) or GPUs [29].

2.5.2. System-Level Techniques

Techniques such as pipelined processing, dual-DSP scheduling, and efficient memory management (e.g., using ping-pong buffers) are employed to ensure that the high computational load does not impede performance. Studies have shown that such strategies can lead to significant speedups, enabling real-time processing even on resource-constrained platforms.

2.6. Rendering, Output, and Post-Processing

2.6.1. Final Adjustments

After stitching, further post-processing might include color correction, exposure adjustments, and fine-tuning of the seam blending to ensure overall visual harmony.

2.6.2. Encoding and Export

The final stitched video is encoded using standard video compression techniques and exported in the desired format. This final step prepares the output for practical applications, whether for live broadcast, immersive viewing experiences (like 360° videos), or archival purposes.

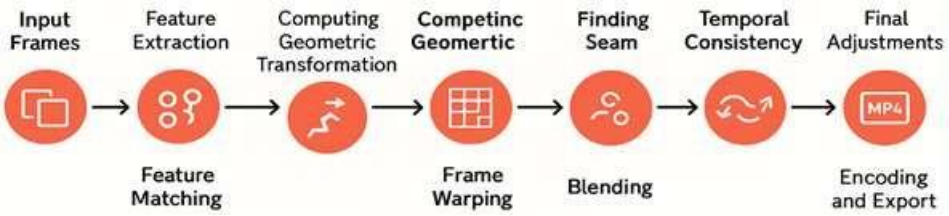

Figure 27: Workflow of video stitching.

3. Applications of Video Stitching

Having explored the detailed pipeline behind video stitching, it is important to understand the diverse and impactful applications where this technology is utilized. The ability to merge multiple video streams seamlessly into panoramic or 360-degree visuals has transformative potential across various fields, ranging from entertainment and virtual reality to surveillance and autonomous systems. This section discusses key application areas,

demonstrating how video stitching is revolutionizing industries and enhancing user experiences in numerous domains.

3.1. Surveillance

Surveillance systems stitch video from multiple cameras to produce panoramic or multi-perspective views of a monitored area. Wide-area security installations (e.g. airports, border zones) commonly combine overlapping camera feeds into a single seamless view.[34] Panoramic stitching allows a single display to cover an extended environment, aiding operators in panoramic area monitoring or spotlight tracking. Temporal coherence is also important: overlapping moving subjects must align over time so that stitched video does not exhibit jumps or ghosting.

Figure 28: Surveillance stitching example.[41]

3.2. Virtual Reality (360° Video)

Immersive VR content (360° video) is created by stitching together videos captured simultaneously by a camera rig (often arranged spherically) [35]. The resulting spherical panorama lets users look in any direction. Deep stitching methods increasingly power this domain, ensuring that seams between camera views remain unnoticeable even as the viewpoint moves. In VR broadcasting and telepresence, maintaining spatial and temporal consistency in the stitched output is crucial for comfort and realism.

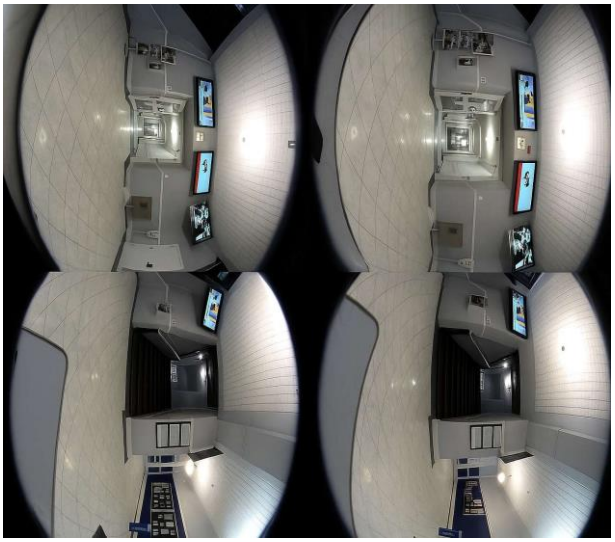

Figure 29: Multi-Camera Input for 360° Video Stitching in Virtual Reality Applications.

3.3. Automotive Surround-View Systems

Modern vehicles use stitched camera views to provide a 360° "bird's-eye" view around the car for parking and collision avoidance. A typical surround-view system employs four fisheye cameras (front, rear, left, right, each ~180° FOV) and stitches their feeds into a cohesive top-down panoramade.mathworks.comvvdntech.com [36]. This stitched panorama greatly reduces driver blind spots and aids tight parking maneuversde.mathworks.com. The approach is becoming ubiquitous: ADAS (advanced driver-assistance) systems now include surround-view as a standard safety.

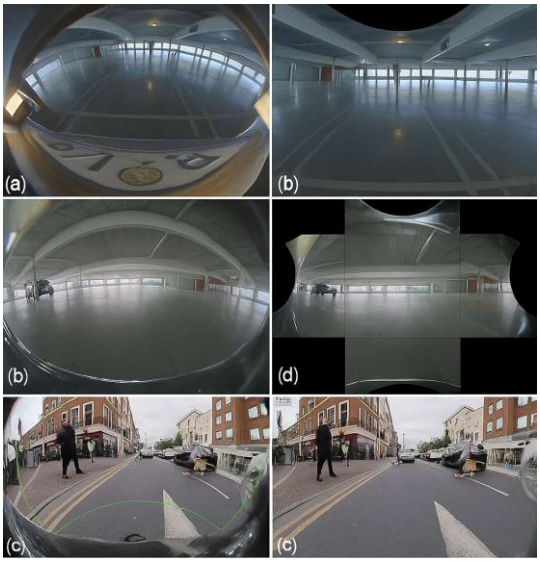

Figure 30: Example of 360° Surround-View Generation from Multi-Fisheye Cameras in Driver-Assistance Systems.

3.4. Sport broadcasting

In live sports (e.g. football, racing, stadium events), stitching enables wide-angle or panoramic replays that capture the entire field or track. Multiple cameras around a stadium can be merged into a single panoramic video or image, giving broadcasters and viewers a comprehensive view of plays.

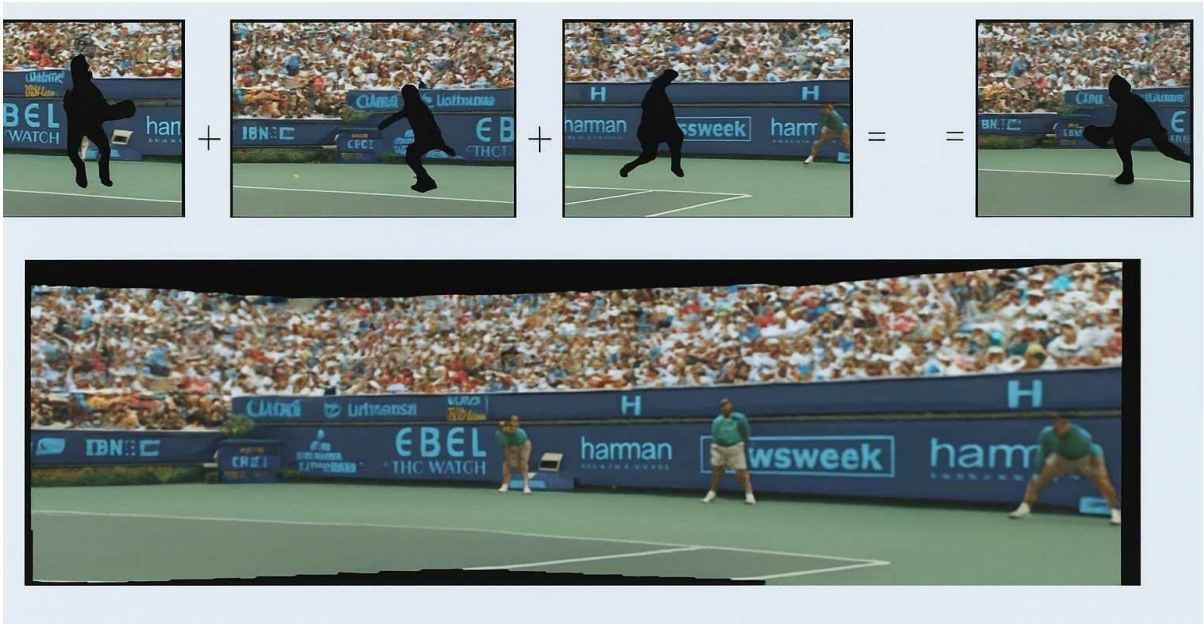

Figure 31: Panoramic Background Stitching for Sports Replay Applications.

3.5. Drones and aerial mapping

Drones capture overlapping images/videos of landscapes (farms, cities, disaster areas) that are then stitched into large-scale maps or 3D models. By aligning successive aerial frames, stitching produces orthomosaic maps or panoramic skyboxes used in agriculture, construction, and geospatial analysis. This “drone mapping” extends the same principles as street-view or satellite stitching: it merges multiple viewpoints into a single coherent map covering a vast area.

Figure 32: Stitched Aerial Panorama from Overlapping Drone Captures.

4. Traditional vs Deep Learning Methods in Video Stitching

Video stitching methods have evolved from traditional computer vision techniques to more advanced deep learning architectures, each offering unique strengths and limitations depending on the application context [7].

4.1. Traditional Methods

Traditional Methods emerged prominently in the late 1990s to early 2000s, based on feature detection and geometric transformations. Key milestones include:

- ***SIFT (1999)*** – Introduced robust keypoint detection invariant to scale and rotation.
- ***RANSAC*** – Used for robust homography estimation and outlier rejection.
- ***Multi-band blending (Burt & Adelson, 1983)*** – Addressed seam smoothing.
  These methods follow a modular pipeline involving detection, matching, warping, and blending, and are widely used in real-time and resource-constrained environments.

4.2. Deep Learning Methods

Deep Learning Methods gained traction from 2016 onward, leveraging data-driven models that learn to extract features, estimate transformations, and blend frames:

- ***Deep Homography (2016):*** A CNN-based method to estimate image alignment without hand-crafted features.
- ***Optical Flow Networks (e.g., PWC-Net):*** For handling motion dynamics between frames.

- ***GAN-based blending and temporal RNNs/Transformers:*** For seamless, temporally coherent outputs.

These approaches excel in handling complex scenes, dynamic motion, and parallax but require significant computational resources and training data [7].

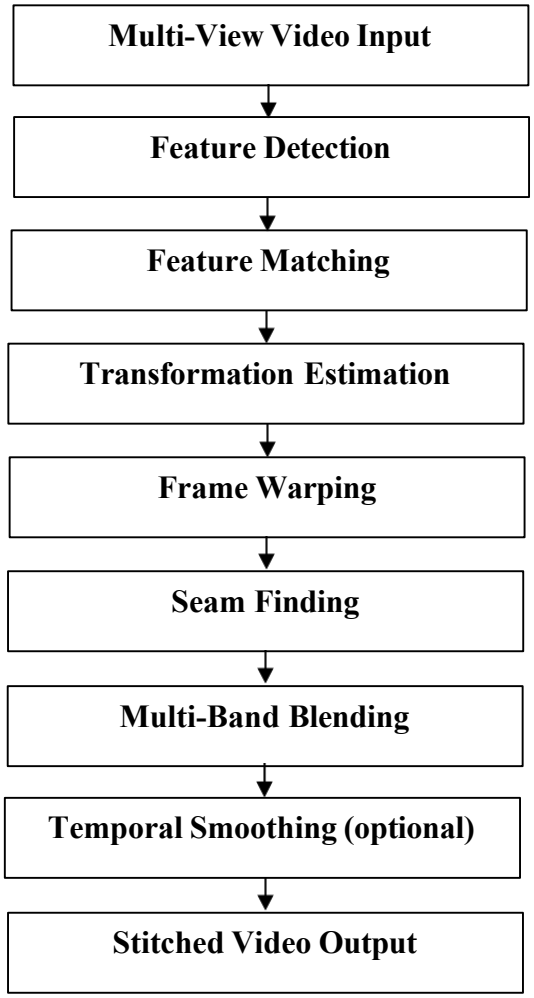

Diagram 1: Modular Pipeline for Video Stitching Using Traditional Computer Vision Techniques.

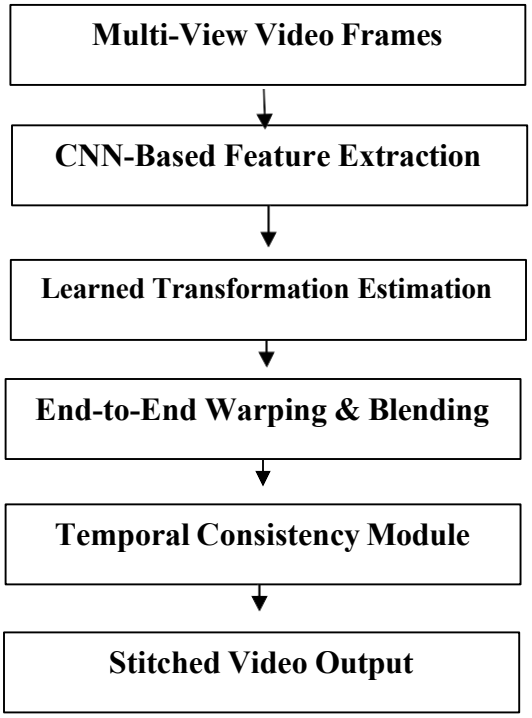

Diagram 2: End-to-End Video Stitching Pipeline Based on Deep Learning Architectures.

To provide a clearer picture of the trade-offs between traditional and deep learning-based approaches, the following comparative table summarizes key aspects such as performance, adaptability, and implementation complexity.

Table 3: Comparative Analysis of Traditional vs. Deep Learning-Based Stitching Methods.

Criterion	Traditional Methods	Deep Learning Methods
Feature Extraction	Hand-crafted (SIFT, SURF, ORB, FAST)	Learned features via CNNs
Transformation Estimation	Geometric (Homography via RANSAC)	Predicted via neural networks (e.g., DeepHomography)
Motion Handling	Limited (works best with static/planar scenes)	Robust to motion and parallax (optical flow, temporal models)
Temporal Consistency	Not explicitly modeled → may cause jitter	Modeled using RNNs, temporal filters, or spatio-temporal learning
Scene Adaptability	Low – sensitive to occlusions, illumination, motion	High – adaptive to complex scenes and non-linear distortions
Blending Techniques	Feathering, multi-band blending	GAN-based blending, seam refinement networks
Hardware Requirements	Moderate (CPU, lightweight GPU)	High (requires modern GPUs and optimization)
Real-Time Performance	High well optimized for real-time	Moderate to low (improving with pruning and acceleration)
Ease of Implementation	Straightforward, modular pipeline	Complex architecture, needs ML frameworks
Training Data	Not required	Required (can be large and annotated)
Explainability	High – transparent steps and logic	Lower – black-box nature in deep networks
Robustness in Complex Scenes	Limited – errors with parallax, dynamic motion	Strong – learned from diverse scenarios
Scalability (multiple inputs)	Linear with camera count	High with parallelism, but complex to train
Typical Applications	Classical panoramas, real-time systems, embedded setups	VR, drone footage, live broadcasting, unstructured inputs
Integration Flexibility	Easy to integrate in existing pipelines	May require custom integration or system redesign

While these methods both traditional and learning-based provide powerful tools for video stitching, their effectiveness is often constrained by real-world challenges that go beyond algorithmic design. The following section explores the key challenges that are unique to video stitching.

5. Challenges unique to video stitching

Video stitching is a natural extension of image stitching applied to temporal sequences. However, while static image stitching deals only with spatial alignment, video stitching introduces additional complexity due to the dynamic nature of video data. This includes temporal coherence, moving elements, and real-time processing constraints. The following challenges are specific to video stitching and must be addressed to ensure a seamless stitched video output.

5.1. Temporal Consistency

One of the most critical challenges in video stitching is maintaining temporal coherence across frames. Even small inconsistencies in alignment or brightness that go unnoticed in static images can result in flickering, blurring, or visual jitter when viewed over time. This is particularly problematic when camera motion is involved or when objects cross seams between video streams [22].

To maintain visual stability, stitching algorithms must ensure that the transformations and blending applied to one frame remain consistent in subsequent frames. This requires tracking not just spatial relationships but also the temporal evolution of keypoints, seams, and motion fields. Inconsistency in these parameters can lead to distracting artifacts that degrade user experience, especially in 360° videos and virtual reality (VR).

5.2. Moving Objects

Video footage typically includes dynamic elements, such as pedestrians, cars, animals, or flowing water. When these moving objects appear in the overlapping regions of two or more video streams, they are often captured at slightly different positions due to parallax or timing differences. If not correctly handled, this results in ghosting artifacts where multiple semi-transparent versions of the same object appear—or tearing, where the object is visibly cut or duplicated at the stitching seam [22].

To address this, advanced techniques such as foreground segmentation, motion detection, and optical flow estimation are integrated into stitching pipelines. This help identify moving regions and either exclude them from alignment computations or apply specialized blending strategies to preserve object continuity across views.

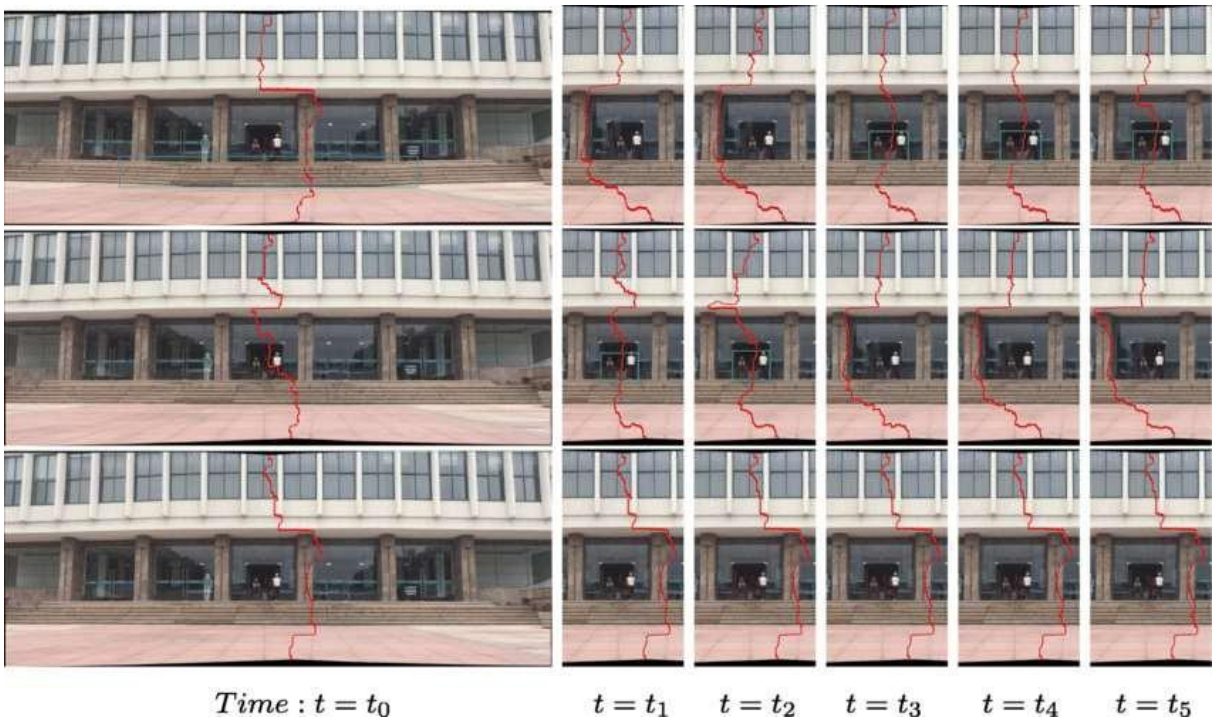

Figure 33: Seam Adjustment Over Time to Preserve Foreground Motion Consistency.

5.3. Camera Motion and Synchronization

In multi-camera setups, all cameras must capture frames synchronously. Even minor desynchronization on the order of milliseconds can cause frames to misalign temporally, especially in fast-motion scenes, leading to visual drift or ghosting [12]. In systems where synchronization hardware is not available, timestamp alignment and software-based frame adjustment techniques must be used.

Furthermore, camera motion introduces its own set of issues. For example, in drone-mounted or handheld rigs, vibration or unintentional shifts can distort alignment between frames. These require the integration of video stabilization algorithms that either correct the motion globally or compensate for local warping.

5.4. Real-Time Processing Constraints

Video stitching in real-time environments—such as live streaming, surveillance, or immersive VR—demands extremely fast processing speeds. For instance, stitching 30 or 60 frames per second across multiple streams can impose a significant computational burden. Any delays or dropped frames directly affect the viewing experience, causing lag, frame skipping, or even pipeline failure.

To meet real-time constraints, algorithms must be optimized for parallel computation and often implemented on GPUs or specialized hardware accelerators. Trade-offs between quality and performance must also be carefully managed, especially in mobile or embedded platforms.

5.5. Exposure and Color Inconsistencies Over Time

Video often captures changing lighting conditions—caused by moving clouds, artificial lighting, or camera auto-exposure adjustments. If different cameras react differently to these changes, their outputs may vary in brightness, contrast, or color tone, causing visible seams or flickering over time in the stitched result.

To handle this, stitching systems must implement temporal exposure correction and color balancing that not only harmonize frames at a given instant but also maintain consistent appearance across time. Techniques such as gain compensation, histogram matching, and temporal smoothing are typically employed.

5.6. Optical Flow and Seam Drift

In static image stitching, seams can be carefully selected and remain fixed. In video, however, scene motion or camera motion can make fixed seams unsuitable over time. This results in seam drift, where previously optimal seam locations become misaligned with content, leading to visual artifacts like edge doubling or object cut-off [22].

To counteract this, modern stitching systems use dynamic seam selection, where seam locations are continuously re-evaluated using optical flow, gradient consistency, and temporal smoothness criteria. These approaches attempt to ensure that seams follow regions with low visual activity and adapt gracefully to motion.

Video stitching introduces a set of unique challenges that require advanced solutions beyond those used in static image stitching. The need for temporal coherence, synchronization, dynamic scene handling, and real-time responsiveness makes video stitching a significantly more complex task. Addressing these challenges is crucial for generating high-quality panoramic or immersive video experiences, particularly in demanding applications such as virtual reality, surveillance, and live broadcasting.

Having outlined the major technical challenges specific to video stitching, the following section reviews recent methods from the literature that attempt to address these issues through both traditional and learning-based approaches.

6. Literature Review and Method Comparison

In this section, we review recent video stitching methods by grouping them according to the key challenges they aim to solve. Rather than a chronological overview, this thematic organization enables a clearer comparison of how different approaches address core issues such as temporal inconsistency, parallax handling, real-time performance, and stitching from unstructured or mobile inputs.

- **Temporal Instability and Warping Shake**

Nie et al. [13] introduced StabStitch++, an unsupervised learning framework targeting the often-overlooked issue of "warping shake" — temporal jitter caused not by camera motion, but by inconsistently warped frames. Unlike prior systems that treat stitching and stabilization separately or via costly joint optimization, StabStitch++ unifies spatial and temporal warps with a warp smoothing model trained end-to-end. Notably, it leverages a novel bidirectional decomposition of homographies on a virtual midplane and achieves real-time performance without requiring synchronized or stabilized source videos, thus significantly improving visual stability and reducing distortions in online settings.

- **Parallax and Wide Baseline Alignment**

Lai et al. [16] proposed a Pushbroom Stitching Network to address the severe parallax and occlusion issues inherent in wide-baseline camera arrays, as encountered in autonomous driving or immersive telepresence. Inspired by the physical model of pushbroom cameras, their approach interpolates spatial views between cameras using learned optical flow and a dedicated interpolation layer. This method avoids ghosting and object breakage, ensuring both

spatial and temporal continuity across frames, a significant advance over conventional homography-based or mesh-based stitching pipelines.

- **Spatio-Temporal Consistency with Moving Foregrounds**

Jiang and Gu [17] tackled the intertwined problems of alignment and composition in dynamic scenes by proposing Spatial-Temporal Content-Preserving Warping (STCPW). This method applies local warping across both spatial and temporal dimensions, enabling accurate alignment even when foreground objects move across views. Seams are computed via a 3D graphcut that factors in object and motion saliency, preventing artifacts from slicing through salient structures. Their results outperform both traditional image-based stitching extensions and commercial software, particularly under high-parallax, non-ideal capture conditions.

- **Unstructured Camera Arrays and Boundary Rectangling**

Pan et al. [14] addressed the problem of stitching videos from unstructured camera arrays, which introduces challenges due to irregular camera placement and mesh inconsistencies. They proposed a two-step optimization approach: first, stitch and rectify a keyframe using mesh warping, then propagate this warp to neighboring frames via global energy minimization. This ensures temporal coherence, rectangular boundaries, and content preservation. Their method is efficient and scalable, suitable for semi-structured multi-camera systems often found in user-level VR rigs or surveillance networks.

Recent work has also explored cross-cutting methods that combine robustness with learning-based refinement. For example, hybrid architectures integrate CNN-based feature descriptors, cost volume matching, and seam adaptation layers to manage low-texture regions and extreme motion. These strategies are often embedded within the core of StabStitch and Pushbroom architectures, further improving performance on real-world datasets.

Together, these studies highlight a clear trend in video stitching: from classical feature matching pipelines toward data-driven, temporally aware, and application-specific methods. This evolution reflects a growing demand for scalable, high-quality video stitching systems capable of adapting to both structured and unstructured environments.

To synthesize these contributions and facilitate a clearer comparison, the following table summarizes key recent methods in video stitching. Each method is categorized by the

primary challenge it addresses, the core approach it employs, and its respective strengths, limitations, and application domains.

Table 4: Comparative Overview of Recent Video Stitching Methods and Their Application-Specific Challenges.

Method	Key Challenge Addressed	Technique Used	Strengths	Limitations	Typical Applications
StabStitch++ [13]	Temporal instability, Warping shake	Bidirectional warping + temporal smoothing with hybrid loss	Real-time, stable, unsupervised, handles unsynchronized input	May underperform with highly dynamic or extremely low-texture scenes	Real-time video stitching for handheld or unsynchronized camera inputs
Pushbroom Stitching Network [16]	Parallax in wide-baseline linear camera arrays	Optical flow-based interpolation via pushbroom layer	Minimizes ghosting and cut-off, handles parallax robustly	Limited to structured setups; may require calibration or known alignment	Autonomous driving, immersive VR, and surveillance with wide-baseline cameras
STCPW [17]	Spatio-temporal consistency with moving foregrounds	Spatial-temporal content-preserving warping + 3D graphcut seams	Handles moving objects, avoids cutting salient structures	Computationally expensive due to local warping and 3D graphcut	Sports, surveillance, and events with large moving subjects across views
Rectangling with Unstructured Arrays[14]	Unstructured camera setups and irregular boundaries	Two-step mesh warping and global optimization for frame propagation	Maintains boundary regularity, efficient, scalable	Assumes fixed boundary layout; may not handle strong camera shake	Consumer VR, semi-structured multi-camera rigs, public space surveillance

7. Current limitations and future directions

Despite significant advances in both traditional and deep learning-based video stitching methods, several limitations remain that constrain their deployment in real-world scenarios.

7.1. Current limitations

7.1.1. Lack of generalization

Many deep models (e.g., Pushbroom, StabStitch) are trained on specific setups or environments, which limits their ability to generalize to unstructured or unseen scenarios without re-training.

7.1.2. High computational cost

Deep learning approaches, especially those involving optical flow, mesh optimization, or GAN-based blending, demand substantial GPU resources, limiting real-time deployment on edge devices or embedded platforms.

7.1.3. Sensitivity to dynamic content

While motion-aware models exist, ghosting and seam drift still persist in highly dynamic scenes, particularly with occlusions or fast motion across overlapping regions.

7.1.4. Synchronization dependency

Methods often assume tightly synchronized camera inputs. In the absence of hardware synchronization (as in consumer or crowdsourced video), performance can degrade significantly.

7.1.5. Complex pipelines and integration

Hybrid or learning-based pipelines require multiple modules (feature extraction, flow estimation, blending), increasing integration complexity and reducing explainability.

7.2. Future directions

As video stitching continues to gain relevance across domains such as immersive media, autonomous systems, and intelligent surveillance, research is evolving toward more integrated, intelligent, and adaptable solutions. Emerging trends emphasize architectural

unification, data-efficient learning, and deployment-ready models that can operate across diverse real-world conditions. The following directions highlight promising avenues for advancing the field, organized into three thematic areas: architectural innovations, learning strategies, and deployment robustness.

7.2.1. Architectural Innovations

- **Unified end-to-end architectures**

  Future research may benefit from fully integrated models capable of learning alignment, warping, and blending jointly with temporal consistency enforcement.

- **Dynamic seam prediction and adaptive blending**

  Incorporating attention mechanisms or content-aware seam adaptation can help maintain stitching quality even in highly variable or low-texture regions.

7.2.2. Deployment & Robustness in Real-World Environments

- **Lightweight, real-time models**

  There is a growing need for optimized, pruned, or quantized models that can deliver acceptable quality on mobile hardware or in live streaming contexts.

- **Cross-domain robustness**

  Developing methods that adapt across domains (e.g., indoor vs. outdoor, day vs. night) will enhance the applicability of video stitching in heterogeneous environments.

7.2.3. Learning Strategies & Data Efficiency

- **Self-Supervised and transfer learning approaches**

  Reducing the dependency on annotated datasets through unsupervised learning (e.g., warping loss, temporal constraints) could improve scalability.

Conclusion

This chapter provided a structured overview of video stitching, from its core principles to the challenges and recent advancements in the field. Traditional methods offer simplicity and real-time performance but struggle with dynamic scenes and complex motion. Deep learning approaches, while more powerful and adaptable, come with higher computational demands and integration complexity.

By comparing recent methods, we highlighted how current research addresses key issues like temporal inconsistency, parallax, and synchronization. However, limitations remain in terms of generalization, scalability, and real-world applicability.

In contrast, the next chapter presents the implementation of our proposed system, which adopts both simple and advanced video stitching techniques. The focus is on balancing practicality, efficiency, and visual quality, offering fast processing with minimal complexity while ensuring high-quality outputs through advanced methods when needed.

Chapter III

Implementation

Introduction

This chapter presents the practical implementation of the video stitching techniques discussed in the previous chapters. The focus is on translating theory into working systems by applying both traditional and advanced methods. The objective is to construct an efficient stitching pipeline capable of producing stable panoramic video outputs. Tools such as OpenCV and deep learning frameworks are employed.

- The following input videos was used to evaluate the stitching methods implemented in the application.

Figure 34: Screenshot from left video.

Figure 35: Screenshot from right video.

1. Development Environment and Tools

1.1. Hardware and Software Setup

1.1.1. Visual studio code

Visual Studio Code, commonly referred to as VS Code is an integrated development environment developed by Microsoft for Windows, Linux, macOS and web browsers. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded version control with Git. Users can change the theme, keyboard shortcuts and preferences, as well as install extensions that add functionality.

VS Code also allowed for seamless switching between script-based execution and interactive exploration, making it an essential tool for code modularization, optimization, and integration with version control systems like GitHub.

Figure 36 and Figure 37 provide visual references to the environments used in this study. Figure 36 illustrates the Visual Studio Code interface where experiments were conducted, while Figure 37 shows the official logo of Python, the programming language used throughout the implementation.

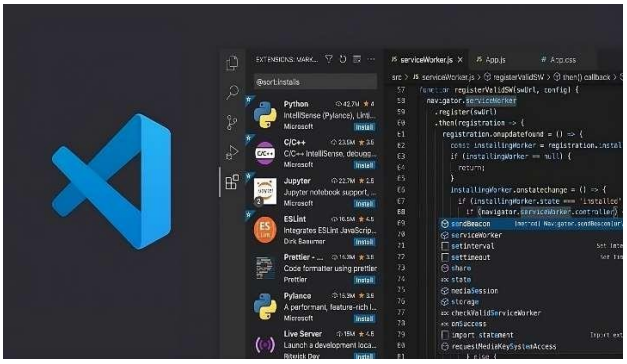

Figure 36: Visual Studio Code interface.

Figure 37: Python Programming Language Logo.

1.2. Programming Libraries and Frameworks

This project was implemented using a focused selection of open-source libraries to handle the entire video stitching pipeline, from GUI design to core image processing and evaluation. The main tools and libraries used were:

1.2.1. GUI and Application Framework

- **PyQt6**: Used to develop the modern, interactive desktop interface. It enabled the creation of custom widgets, layouts, buttons, drag-and-drop zones, and dialog windows with full styling support through Qt stylesheets.
- **QThread & pyqtSignal**: Used for running video processing tasks in the background to maintain a responsive GUI during long computations.

1.2.2. Image and Video Processing

- **OpenCV**: The backbone of video stitching logic. Used for frame capture, feature detection (AKAZE and ORB), keypoint matching (BFMatcher), homography estimation, image warping, blending, and video writing.
- **NumPy**: Used for numerical operations such as entropy calculation and data manipulation during frame processing.
- **OS & sys**: Used for file handling, path management, and controlling system-level operations (e.g., launching file explorers).
- **Skimage (SSIM metric)**: Employed to evaluate the similarity between the stitched video and a reference video using the Structural Similarity Index (SSIM).

1.2.3. Evaluation and Post-processing

- **SSIM via scikit-image**: Implemented in a separate thread to provide objective image quality feedback.
- **Custom entropy calculation**: Used to analyze image complexity and help adjust blending zones dynamically if required.

2. Stitching Algorithms and Methods

This section presents the stitching strategies integrated into the application. Two main operational modes were implemented: Easy Stitch for rapid results using a minimal configuration, and Advanced Stitch Options, offering multiple algorithmic approaches for higher precision and customization. These methods were designed to balance speed, robustness, and visual quality across different use cases.

2.1. Easy Stitch

The Easy Stitch mode is designed for fast, one-click operation. It uses the AKAZE feature detector and binary descriptor, combined with Brute-Force Matching (BFMatcher) and RANSAC-based homography estimation. The computed homography is applied to warp one video frame onto the other, and stitching is repeated using the same transformation across all frames.

- **Advantage:** Fast and reliable for well-aligned or semi-static video sequences.

- **Limitation:** Lacks automatic quality control or adaptive corrections.

2.2. Advanced Stitch methods

The advanced mode provides the user with a set of more sophisticated stitching strategies, selected via the application interface.

2.2.1. Hybrid Intelligent Stitching System

This method implements a semi-intelligent workflow that combines feature-matching efficiency with runtime drift monitoring:

- **Stitch Once Then Warp**: Homography is computed once using ORB + RANSAC and reused for all frames.
- **Drift Monitor**: Compares successive homographies (Ht vs. Ht-1) and re-triggers recalculation if significant drift is detected.
- **Entropy Seam Scanner (optional)**: Calculates entropy in overlapping regions and adjusts seam lines if visual artifacts are detected.

This method offers an optimal balance between stability, processing speed, and visual quality.

2.2.2. Stitching with Attention Maps

An experimental approach using attention mechanisms (e.g., from Vision Transformers) to prioritize semantically or visually significant regions when blending. The goal is to improve seam decisions and preserve key visual content during warping.

- **Advantage:** Scene-aware blending.
- **Note:** This approach requires pre-extracted attention maps or integration with transformer-based models.

2.2.3. Genetic Seam Optimization

Uses a genetic algorithm to find the best seam path in the overlapping region. The algorithm evolves multiple seam candidates based on fitness criteria like edge continuity and color consistency.

- **Advantage:** Produces highly refined seams in complex overlaps.
- **Trade-off:** More computationally intensive than fixed-seam methods.

2.2.4. Seam Inpainting using GAN

Applies Generative Adversarial Networks to correct visual inconsistencies along the seam after stitching. The inpainting model fills the seam area with realistic textures to match surrounding regions.

- **Advantage:** Visually seamless output even with imperfect alignment.
- **Use case:** Ideal for scenes with occlusions or parallax.

3. System Interface

3.1. Video Input Selection

- GUI allows selection of input files (left video and right video):

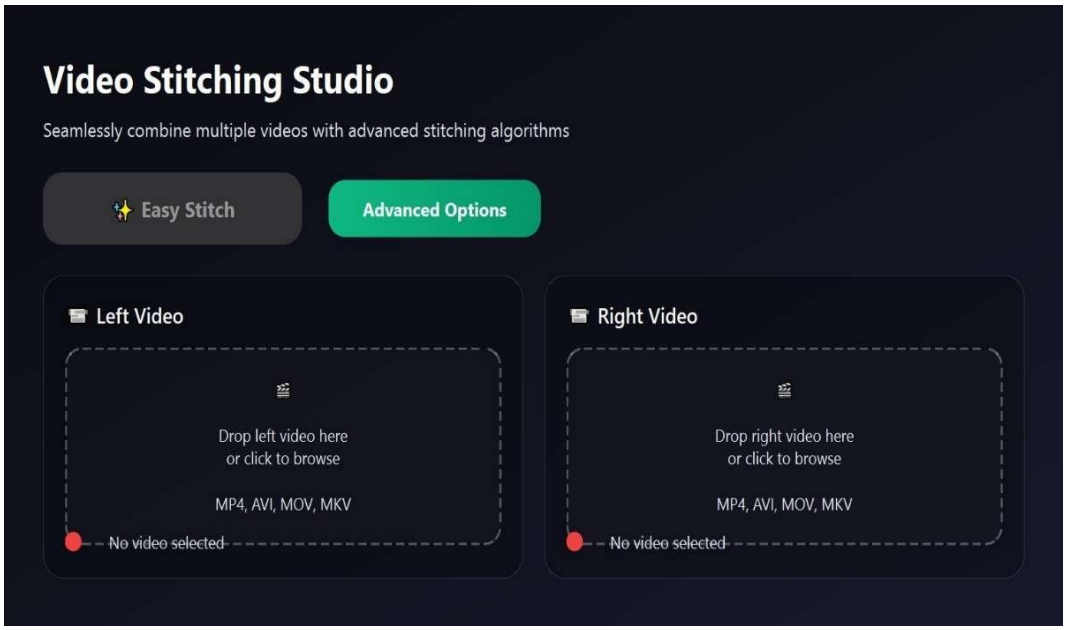

Figure 38: Video input selection interface

3.2. Parameter Configuration

- user can choose between easy stitch method or advanced method.

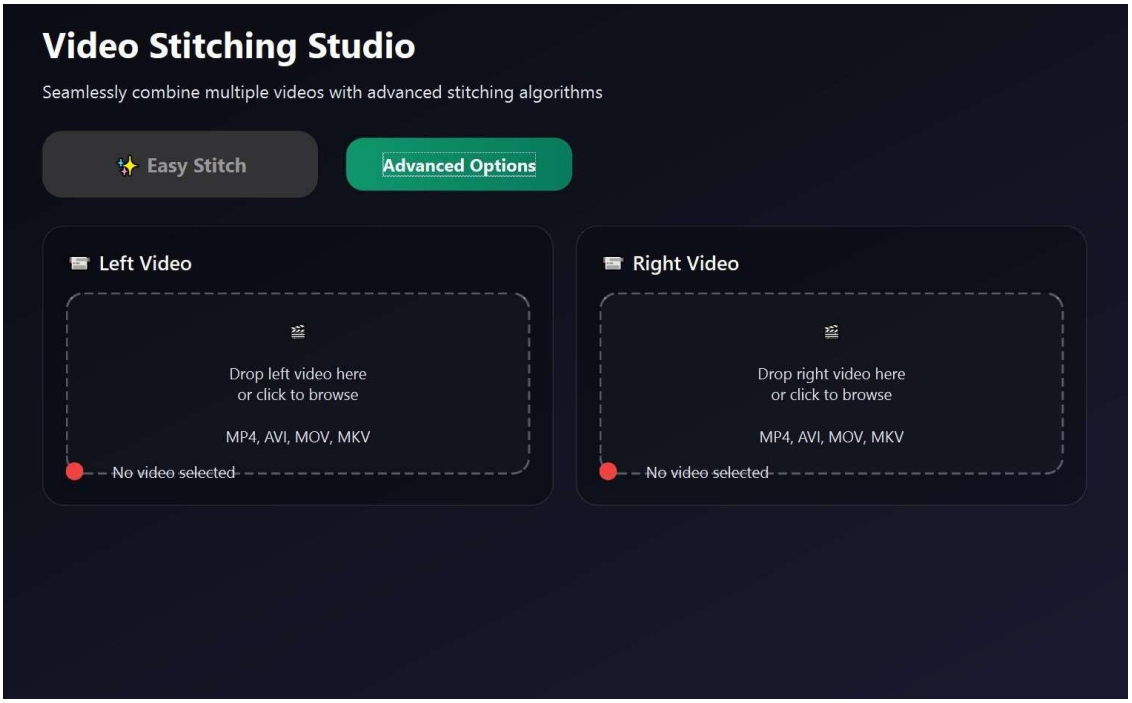

Figure 39: Stitching method selection menu

- Users can switch between advanced stitching methods using the dropdown menu.

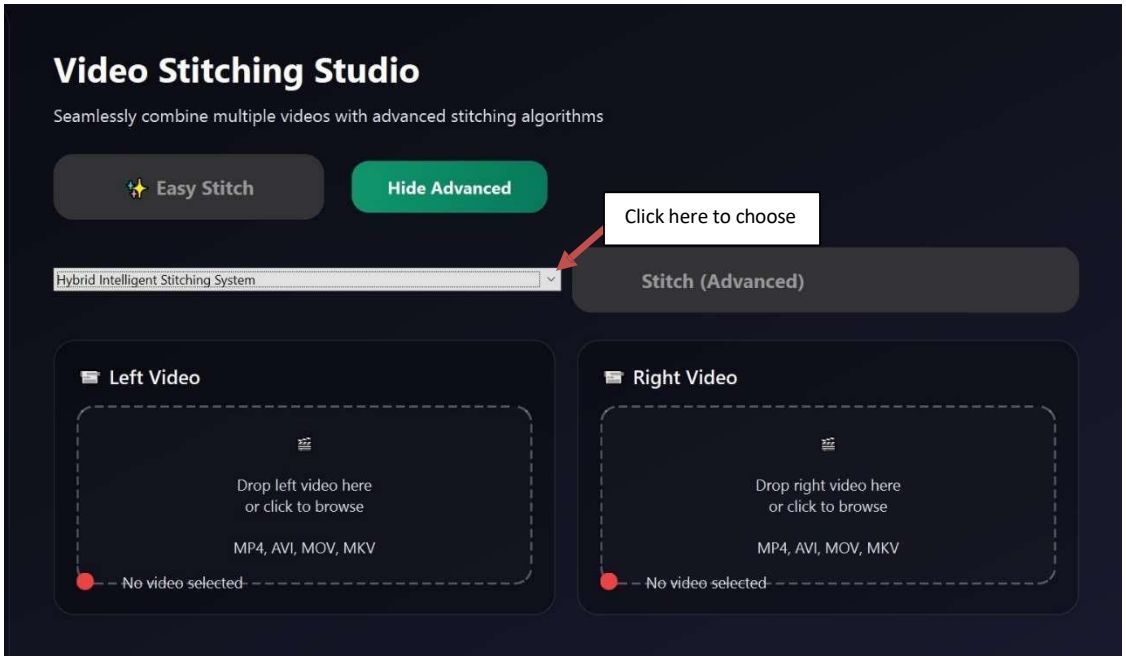

Figure 40: Advanced method configuration options

3.3. Output

The stitched video is saved locally, and users can access it directly by clicking the "Open Folder" button:

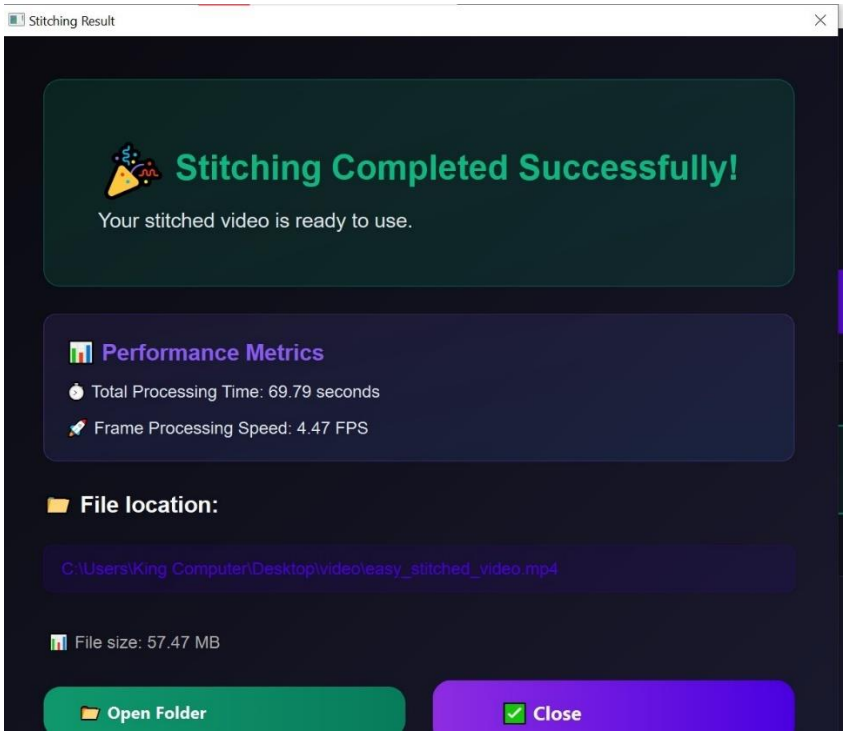

Figure 41: Output management and access

4. Evaluation Strategy and Quality Metrics

To assess the quality of stitched outputs, the application integrates an automated evaluation module based on **SSIM** (Structural Similarity Index) and PSNR.

4.1. Definition of SSIM (Structural Similarity Index)

SSIM is a perceptual metric that measures the similarity between two images based on luminance, contrast, and structure. The index ranges from -1 to 1, where 1 indicates perfect similarity.

4.2. Definition of PSNR (Peak signal-to-noise ratio)

Peak signal-to-noise ratio (PSNR) is an engineering term for the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. Because many signals have a very wide dynamic range, PSNR is usually expressed as a logarithmic quantity using the decibel scale.

4.3. Time and Speed Metrics

Execution time and processing speed were recorded to assess the computational performance of each stitching method under evaluation.

4.3.1. Performance Indicators

The following time-based metrics were measured for each stitching method:

- **Total Processing Time:** The duration from initialization to output rendering.
- **Frame Processing Speed (FPS):** The average number of frames processed per second, computed over the full sequence.

The table below summarizes the performance indicators recorded for each stitching method, including SSIM, PSNR, processing time, and frame rate.

Table 5: Comparative Performance Metrics of Video Stitching Methods

Method	Avg. SSIM	PSNR	Total Time (s)	FPS	Notes
Easy Stitch (AKAZE)	0.873	39	57.90	5.39	Fast, general-purpose baseline
Hybrid Intelligent Stitching	0.891	40	59.3	5.26	Drift-aware, better seam control
Attention Map Guided Stitching	0.894	40	59.1	5.24	Enhanced alignment at key zones
Genetic Seam Optimization	0.886	41	60.7	5.22	Seam refinement via entropy
GAN-based Inpainting	0.879	40	59.5	5.27	Improved aesthetics, slower

Discussion

The experimental results demonstrate that all the evaluated video stitching methods achieve relatively high perceptual quality, with SSIM scores ranging from 0.873 to 0.894 and PSNR values between 39 dB and 41 dB. These metrics indicate a high level of structural and signal fidelity, as values above 0.85 (SSIM) and 35 dB (PSNR) are generally associated with visually satisfactory outcomes in image and video processing tasks.

In terms of computational efficiency, the average frame rate across all methods hovered around 5 FPS, with total execution times varying only slightly between methods. This reflects a consistent baseline in terms of processing load, making all methods feasible for offline or semi-real-time applications.

Method-Specific Insights

- **Easy Stitch (AKAZE)** serves as a fast, general-purpose baseline. With an SSIM of **0.873** and **PSNR of 39 dB**, it provides acceptable visual quality while requiring minimal processing. This makes it suitable for scenarios where speed is prioritized over seam precision.
- **Hybrid Intelligent Stitching** outperformed other methods in terms of perceptual quality, achieving an **SSIM of 0.891** and **PSNR of 40 dB**. Its dynamic homography update and entropy-based seam detection help minimize geometric drift and temporal inconsistency, offering an excellent trade-off between robustness and computational cost.
- **Attention Map Guided Stitching** achieved the **highest SSIM score (0.894)**, indicating that its region-aware blending strategy effectively preserves structural coherence. This method is especially well-suited for scenes with critical visual details near stitching seams.
- **Genetic Seam Optimization** yielded the **highest PSNR (41 dB)**, suggesting superior handling of pixel-level color and gradient continuity. However, its SSIM score, while high (**0.886**), was marginally lower than the attention-guided approach, likely due to its pixel-centric rather than structure-centric optimization.

- **GAN-based Inpainting** achieved good perceptual metrics (**SSIM: 0.879**, **PSNR: 40 dB**) and performed well in suppressing seam artifacts. While not the top performer in terms of scores, it offers the best visual aesthetics when dealing with irregular or visually inconsistent seam regions, at the cost of slightly more computational load.

General Observations

- The **difference in quality across advanced methods is narrow**, indicating that all approaches are effective in maintaining high similarity to the original content.
- **Trade-offs between visual quality and complexity** are evident: while Easy Stitch is faster and simpler, advanced methods provide marginal improvements in quality through more sophisticated logic.
- **Entropy-based seam refinement** (used in Hybrid and Genetic methods) consistently improves seam coherence and blending, as reflected in both SSIM and PSNR scores.

In conclusion, method selection should depend on the application context. For quick tasks, Easy Stitch suffices. For high-fidelity outputs in critical applications (e.g., surveillance, film post-production), **Hybrid** or **Attention Map Guided Stitching** offers the best balance of performance and quality.

Conclusion

This chapter detailed the implementation and evaluation of various video stitching methods, highlighting both traditional and advanced approaches. The system was developed with a focus on efficiency, visual quality, and flexibility, offering both quick stitching via the Easy Stitch method and more refined outputs through advanced techniques like Hybrid Intelligent Stitching, Attention Map Guided Stitching, Genetic Seam Optimization, and GAN-based Inpainting. The evaluation demonstrated that while all methods provided high-quality results, the choice of method depends on the specific application needs, such as processing speed or visual fidelity. For quick tasks, the Easy Stitch method proved to be sufficient, whereas the advanced methods offered improvements in seam accuracy and visual coherence, particularly suited for professional-grade applications. The results confirm the system's capability to produce stable and high-quality panoramic video outputs, with a reasonable trade-off between computational load and visual quality.

General Conclusion

This dissertation explored the intricate process of image and video stitching, from fundamental concepts to advanced techniques, with an emphasis on the practical implementation of video stitching methods. Through the detailed analysis in Chapter 1, the historical evolution of image stitching was reviewed, highlighting key milestones that paved the way for modern, more sophisticated methods. The advent of powerful algorithms such as SIFT, SURF, and ORB, and the rise of deep learning models like Deep Homography, have dramatically enhanced the accuracy and efficiency of stitching techniques.

In Chapter 2, we delved deeper into the specifics of video stitching, which brings additional challenges due to the temporal dimension, motion handling, and the need for real-time processing. The video stitching pipeline was examined, with a focus on feature extraction, frame warping, and blending techniques, as well as strategies to maintain temporal coherence and ensure seamless integration across frames. These challenges are particularly crucial in applications such as virtual reality, surveillance, and 360-degree video production, where the stitching must be both precise and real-time.

Chapter 3 provided a practical implementation of these methods, showcasing how the principles discussed earlier were translated into a working video stitching system. This system incorporates various algorithms, including the Easy Stitch method, Hybrid Intelligent Stitching, Attention Map Guided Stitching, and GAN-based Inpainting, each tailored to different application needs. The evaluation of these methods revealed that while all approaches produced satisfactory results, advanced methods such as Hybrid Stitching and Attention Map Guided Stitching offered improved visual quality and stability, particularly in dynamic and complex scenes.

In conclusion, this research successfully integrated theory and practice to advance the field of video stitching. By balancing computational efficiency with visual quality, this study provides a foundation for developing real-time, high-fidelity video stitching systems. The results underscore the importance of selecting appropriate stitching methods based on specific requirements, whether for fast, real-time applications or for professional-grade, high-quality visual outputs.

Bibliography

[1] A. Farhadi, *Image Stitching*. CSE 576 Lecture Notes, Univ. of Washington, Spring 2008.

[2] E. Adel, M. Elmogy, and H. El-Bakry, "Image Stitching based on Feature Extraction Techniques: A Survey," *Int. J. Comput. Appl.*, vol. 99, no. 6, pp. 1–9, Aug. 2014.

[3] V. S. Sakharkar and S. R. Gupta, "Image Stitching Techniques – An Overview," *Int. J. Comput. Sci. Appl.*, vol. 6, no. 2, pp. 324–330, Apr. 2013.

[4] S. Soltanpour and C. Joslin, "A Survey on Feature-Based and Deep Image Stitching," in *Proc. 20th Int. Joint Conf. on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*, 2025.

[5] N. K. El Abbadi, H. A. Wahab, and M. S. Al-Juboori, "A Review Over Panoramic Image Stitching Techniques," *J. Phys.: Conf. Ser.*, vol. 1999, 012115, 2021. DOI: 10.1088/1742-6596/1999/1/012115

[6] W. Lyu, Z. Zhou, L. Chen, and Y. Zhou, "A Survey on Image and Video Stitching," *Virtual Reality & Intelligent Hardware*, vol. 1, no. 2, pp. 125–145, 2019. DOI: 10.1016/j.vrih.2019.03.001

[7] Z. Yang, Y. Yin, H. Xu, Q. Jing, Z. Jiang, T. Liao, and C. G. Soares, "Advancements of Image and Video Stitching Techniques: A Review," *IEEE Sensors Journal*, early access, 2025. DOI: 10.1109/JSEN.2025.3563082

[8] S. K. Sharma, K. Jain, and A. K. Shukla, "A Comparative Analysis of Feature Detectors and Descriptors for Image Stitching," *Applied Sciences*, vol. 13, no. 11, 6015, 2023. DOI: 10.3390/app13116015

[9] K. Joshi, "Approaches and Challenges in Real Time Image Stitching," *Int. J. Sci. Res. Eng. Manag. (IJSREM)*, vol. 4, no. 5, pp. 1–5, May 2020.

[10] M. Lin, T. Liu, Y. Li, X. Miao, and C. He, "Image Stitching by Disparity-Guided Multi-Plane Alignment," *Signal Processing: Image Communication*, vol. 101, 116539, 2022.

[11] O. Jacobsson and T. Talpalar, *Image Stitching for Video: Feature Detection & Optical Flow*, M.Sc. thesis, Dept. of Engineering Physics and Electrical Engineering, 2024.

[12] R. Szeliski, "Image Alignment and Stitching: A Tutorial," *Foundations and Trends® in Computer Graphics and Vision*, vol. 2, no. 1, pp. 1–104, 2006. DOI: 10.1561/0600000009

[13] O. Soler Cubero, *Image Stitching*, Technical Report, June 2011.

[14] H. Niitsuma and T. Maruyama, "Sum of Absolute Difference Implementations for Image Processing," in *Proc. Int. Conf. Field-Programmable Technology (FPT)*, Jan. 2011, pp. 1–4.

[15] M. B. Hisham, S. N. Yaakob, and R. A. A. Raof, "Template Matching Using Sum of Squared Difference and Normalized Cross Correlation," in *Proc. 6th Int. Conf. on ICT for The Muslim World (ICT4M)*, Apr. 2016, pp. 1–6.

[16] T. Lindeberg, "Scale Invariant Feature Transform," *Scholarpedia*, vol. 7, no. 5, p. 10491, May 2012. DOI: 10.4249/scholarpedia.10491

[17] A. K. Tripathi and S. S. Mishra, "A Survey on Motion Detection by Image Stitching Techniques," *International Journal of Computer Applications*, vol. 160, no. 6, pp. 24–29, Feb. 2017. DOI: 10.5120/ijca2017913125

[18] D.-B. Xu, H.-M. Tao, J. Yu, and C.-B. Xiao, "Real-Time Multi-Camera Video Stitching Based on Improved Optimal Stitch Line and Multi-Resolution Fusion," in *Proc. ICIG*, 2017, pp. 357–368. DOI: 10.1007/978-3-319-70136-3_30

[19] A. R. Mishra and A. Baranwal, "Real-Time Image and Video Stitching via Seamless Integration of Live Camera Feeds," *IJARCCE*, vol. 12, no. 10, pp. 45–51, Oct. 2023.

[20] R. Pan, Y. Zhang, L. Xu, A. Qin, and H. Du, "Stitching Videos From Unstructured Camera Arrays With Rectangular Boundaries," *IEEE Access*, vol. 12, pp. 74856–74868, 2024.

[21] P. Baheti, "Virtual Reality Content Creation Technology," *Proc. SPIE Digital Imaging Conf.*, Feb. 2017.

[22] W. Jiang, "Video Stitching with Spatial-Temporal Content-Preserving Warping," Huawei Media Lab, Futurewei Technologies Inc., Tech. Rep., 2015.

[23] L. Nie, C. Lin, K. Liao, Y. Zhang, S. Liu, and Y. Zhao, "StabStitch++: Unsupervised Online Video Stitching with Spatiotemporal Bidirectional Warps," *IEEE Trans. Multimedia*, vol. 14, no. 8, pp. 1902–1915, Aug. 2021.

[24] A. Alian and L. Gierup, *Analysis of Seam Algorithms in Video Stitching with Static Scenes and Dynamic Objects*, Master's Thesis, Chalmers University of Technology, 2024.

[25] C. Xie, X. Zhang, H. Yang, L. Chen, and Z. Gao, "Video Stitching Based on Optical Flow," in *Proc. Int. Conf. on Intelligent Information Technology and Security (ICIITS)*, 2018, pp. 1–5.

[26] H. Min, X. Sun, and Z. Zhang, "Panoramic Video Stitching in Manufacturing Workshops Based on Key Target Protection," in *Proc. IEEE Int. Conf. on Industrial Cyber-Physical Systems (ICPS)*, May 2025, pp. 1–6.

[27] R. Patel, S. Pandey, C. I. Patel, and R. Paul, "Effective Flicker Detection Technique Using Artificial Neural Network for Video," in *Proc. Int. Conf. on Research and Innovations in Science, Engineering & Technology (ICRISET)*, 2017, pp. 1–5.

[28] R. Pan, Y. Zhang, L. Xu, A. Qin, and H. Du, "Stitching Videos From Unstructured Camera Arrays With Rectangular Boundaries," *IEEE Access*, vol. 12, pp. 11196–11210, Feb. 2024, doi: 10.1109/ACCESS.2024.3350723.

[29] D.-B. Xu, H.-M. Tao, J. Yu, and C.-B. Xiao, "Real-Time Multi-Camera Video Stitching Based on Improved Optimal Stitch Line and Multi-Resolution Fusion," in *Proc. Int. Conf. Image and Graphics (ICIG)*, 2017, pp. 357–368, doi: 10.1007/978-3-319-70136-3_30.

[30] "Geometric Transformation in Image Processing," *GeeksforGeeks*, Apr. 17, 2025. [Online]. Available: https://www.geeksforgeeks.org/geometric-transformation-in-image-processing.

[31] W.-S. Lai, O. Gallo, J. Gu, D. Sun, M.-H. Yang, and J. Kautz, "Video Stitching for Linear Camera Arrays," *arXiv preprint arXiv:1907.12532*, Jul. 31, 2019.

[32] R. Szeliski, "Image Alignment and Stitching: A Tutorial," *Foundations and Trends in Computer Graphics and Vision*, vol. 2, no. 1, pp. 1–104, 2006.

[33] H. Guo, S. Liu, T. He, S. Zhu, B. Zeng, and M. Gabbouj, "Joint Video Stitching and Stabilization From Moving Cameras," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5491–5503, Nov. 2016, doi: 10.1109/TIP.2016.2603987.

[34] B. He and S. Yu, "Parallax-Robust Surveillance Video Stitching," *Journal of Visual Communication and Image Representation*, vol. 38, pp. 668–678, Dec. 2015, doi: 10.1016/j.jvcir.2015.12.012.

[35] *"360-Degree Video Stitching Tool,"* Focus Photo and Video. [Online]. Available. [Accessed: 16 Jun. 2025].

[36] Texas Instruments, *"360° Video Stitching Reference Design: SPRY270A,"* Application Report, TI Literature, Mar. 2019. [Online]. Available. [Accessed: 16 Jun. 2025].

[37] "Image Stitching," *SlideServe*, (accessed Jun. 20, 2025).

[38] A. Goretkin, "Image Stitching in Computer Vision," *Baeldung on Computer Science*, (accessed Jun. 20, 2025).

[39] T. Mamić, "Image Stitching," *Mono Software Blog*, Mar. 14, 2018. [Online]. Available: (accessed Jun. 20, 2025).

[40] "BioStitch-500: Live Image Stitching & Digital Slide Scanning Software with Free Camera," *BioImager*, [Online]. Available: (accessed Jun. 20, 2025).

[41] "BioStitch-500: Live Image Stitching & Digital Slide Scanning Software with Free Camera," *BioImager*, [Online]. (accessed Jun. 20, 2025).

[42] S. A. K. Tareen and Z. Saleem, "A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK," Apr. 2018, pp. 1–10.