A Study on Real-time Video Mosaicking
and
Stabilization Using High-speed Vision
（高速ビジョンを用いた
リアルタイムビデオ モザイキングと
安定化に関する研究）

by

Raut Sushil

Graduate School of Engineering
Hiroshima University
March, 2020

Contents

List of Figures

List of Tables

Chapter 1

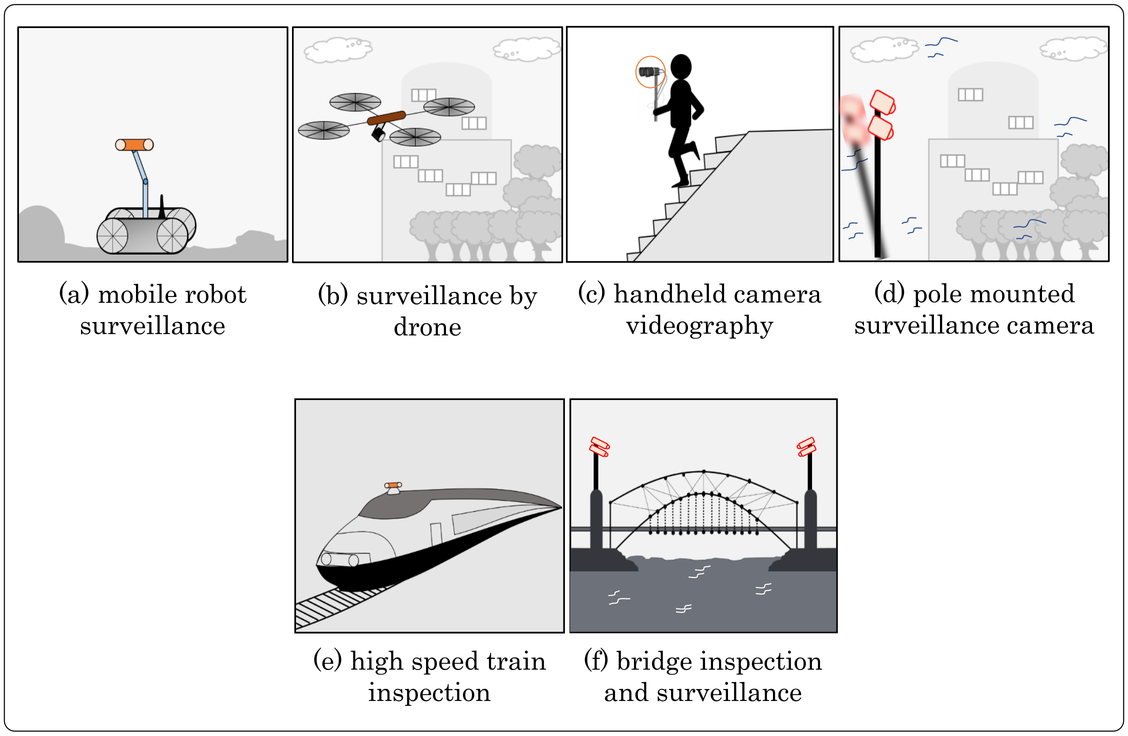
Introduction

In this chapter, we explain the problems associated with digital photography and videography, which have been observed in various applications. Mainly we explain the related works on feature tracking, displacement and motion anomalies estimation methods along with the broad scope of panoramic and smooth video composition. The studies on various applications of high-speed vision systems in object tracking, mosaicking, and in numerous fields are also explained in this chapter. We also proposed a high-frame-rate (HFR) vision-based displacement and motion anomalies sensing method to overcome the shortcomings of conventional systems.

1.1 Research Background

1.1.1 Digital Photography and Videography

The digital-camera-technologies have been advancing day-by-day to deliver feasible photographic and videographic functionalities in commercial cameras. Image processing functionalities such as panorama, color filtering, noise filtering, etc. have become onboard functions of the modern generation digital cameras to provide run-time solutions to the users. Digital Cameras furthermore featured with image stabilization facilities, which also called vibration reduction and gimbals compensation. Stabilization functionality not only demanded by handheld cameras but also explicitly required for mobile robots, uneven-terrain vehicles as well as modern Unmanned-Aerial-Vehicles (UAV) to obtain smoothed video sequences. As shown in Figure 1.1, digital photography, and videography have a broad scope. High-Resolution image sequences are suitable for capturing wide information of the scenic world for monitoring purposes as well as minute details of the microscopic world for scientific study. It is also obligatory in most of the scenarios where video quality of captured sequences is affected by induced undesired motions.



(a) mobile robot surveillance

(b) surveillance by drone

(c) handheld camera videography

(d) pole mounted surveillance camera

(e) high speed train inspection

(f) bridge inspection and surveillance

Figure 1.1: Scope of Panoramic Views and Smooth Videos in Digital Photography and Videography

The undesired motions are introduced by various means of vibration sources such as, uneven terrains induce vibration in videos taken by camera mounted on mobile robot for surveillance purpose Figure 1.1-(a), high-speed wind and propeller rotation produce a vibration in videos captured by camera mounted drone while flying Figure 1.1-(b), videos captured by handheld camera usually affected by walking on stairs and also due to uneven motions of operator Figure 1.1-(c), pole-mounted terminal-cameras often capture shaky videos due to vibration induced by high-speed wind or seismic activities Figure 1.1-(d), camera mounted on high-speed trains usually captures videos that are affected by periodic mechanical vibrations Figure 1.1-(e), moreover, cameras appointed for structure health monitoring also get influenced by structural vibration due to seismic or human-induced fluctuations Figure 1.1-(f).

The smooth visibility of the captured sequences is usually affected by undesired motions caused due to rapid apparent movements between camera and scene. The changes while rolling shutter becomes shakiest when robots or vehicle moves on uneven surfaces. So it is required to compensate or neutralize undesired motions by filtering it from the desired action. Video stabilization is a process of removing irregular shaky-effects by correcting captured sequences. It eliminates global motion anomalies by altering affected images sequences. Professional videographers and researchers who are working in the domain of digital image processing and machine learning have reported the problem of motion blur and jittery videos due to the unsteadiness of the operators. Video captured by a handheld camera or body mounted camera during sports such as ski bordering, skating, running, hiking, and other games where continuous rapid motions are happening, jitter always affects the smoothness of captured videos.

Machine learning and digital image processing have been also suffered the problem of losing vital information due to jitter that results in unreasonable noise. The cameras used for mobile and teleoperated robotics are assumed as the most informative sensors. When they rapidly maneuver on non-uniform surfaces, random vibration introduces in the captured videos, which limits decision making ability at known or unknown scenarios. Tele-operated robots have a limitation of video transmission bandwidth on wireless networks. That means there is no scope to stabilize images instead of synthesizing at the terminal camera. In the case of vision-guided vehicles, conventionally light detection and ranging (LiDAR) along with a series of cameras used as prime sensors to localize themselves in the surrounding environment. In their acquired videos, the oscillatory motion components are often observed. In this case, jitter becomes a dominating noise that may tend to misguiding or lost location. The emerging mobile surveillance technology such as aerial vehicles, UAVs are used for aerial photography, surveillance, navigation, localization and tracking purposes. They often track suspected activities observed in targeted environments via high-resolution camera-based tracking systems. UAVs surveillance is a significant advantage during disaster situations. Smoothly captured videos are a trade-off for human operators in rescues operations. During crucial moments, real-time video footage is highly demanded not to miss landmarks nor target recognition. UAVs with conventional camera systems often fail due to defocusing and blurriness in the videos caused by mechanical vibration while flying at high altitudes and capturing optically or digitally zoomed scenes. The widely used wireless technologies in the surveillance cameras are also exposed to undesired noise due to high-speed wind or structural vibrations.

1.1.2 Conventional Video Mosaicking and Stabilization

1.1.2.1 Video Mosaicking Related Works

The image mosaicking is an image registration technique in which image sequences captured at different time intervals are merged into a single composite image that covers a larger, more seamless view than the field of view of the camera. It has been used to acquire panoramic pictures for natural scenes in several applications, such as surveillance, augmented reality, remote sensing, and digital camera panorama generation. In the past decades, many image mosaicking approaches have been proposed for accurate image registration of overlapping images. However, most of these proposed approaches perform offline because of the heavy computational loads they incur in trying to optimize and achieve global consistency in panoramic images.

Image mosaicking is a standard technique in computer vision; many image registration methods for image mosaicking have been proposed [1,2]. To produce panoramic images for post-visualization after capturing the whole image sequence, most of these methods employ offline image mosaicking computation, which uses pairwise image matching and then global optimization to ensure consistency. These image registration methods can be roughly categorized into region-based methods [3–5] and feature-based methods [6–9]. Region-based methods can provide accurate image registration using all the available image data; however, they incur a heavy computational load and are oversensitive to image brightness. In feature-based methods, the transform parameters between images are determined by calculating feature points using feature extractors such as the KLT tracker [10] and SIFT features [11] for feature point correspondences between images. They can reduce the computational complexity in image registration; however, the outliers in feature point correspondence increase when the feature points are not stably extracted.

Nowadays, the panoramic image mosaics are required not only for post panoramic visualization but also for online panoramic display in many computer vision applications such as motion detection and tracking [12, 13], mosaic-based localization [14], and augmented reality [15]. Consequently, many real-time video mosaicking approaches for fast image registration have been proposed. Kourogi et al. proposed a fast image registration method that uses pseudo motion vectors estimated as gradient-based optical flow [16]. Civera et al. developed a drift-free feature-based image mosaicking system that obtains a consistent image from images of 320×240 pixels at 30 fps using an EKF-SLAM approach when previously viewed scenes are revisited [17]. Botterill et al. realized real-time feature-based image mosaicking for aerial images of 800×532 pixels at 6.2 fps [18]. de Souza et al. performed real-time feature-based image mosaicking without over-deformation of mosaics using a non-rigid deformation model [19].

The standard video signal formats restrict most of the video cameras used in these approaches to dozens of frames per second. However, a camera that is functioning at a low frame rate must be operated slowly to prevent difficulties in matching feature correspondences between frames, because the computational cost and the number of mismatched pairs of feature points increase when image displacements between frames increases. To reduce the mismatch problem in feature point correspondence, caused by large image displacements between frames when a panoramic scene is observed from a fast-moving camera, several groups are currently researching accurate image mosaicking using HFR videos for fast camera motion such as image mosaicking with a flying camera [20]. However, these research efforts are focused on offline HFR videos. If HFR videos could be simultaneously mosaicked to obtain a panoramic image that covers a larger seamless view, real-time image mosaicking could be utilized in many scenarios involving rapid camera motion, such as image mosaicking using a freely operated hand-mobile camera and image mosaicking from fast-moving automobiles.

1.1.2.2 Video Stabilization Related Works

Video stabilization [21–25] is a well-known process that has been implemented to reduce undesired motion in image sequences which occur due to shaking or jiggling of a camera or rapidly moving objects while rolling the shutter. These motion anomalies are jitters caused by various external sources responsible for the shaking of the camera and lead to unpleasant visual effects in video sequences. Sources such as unsteady handling of the camera by an operator, rapidly moving sports camera, or camera-mounted vehicles or robots when maneuvering on uneven surfaces are responsible for jittery motion anomalies. As shown in Figure 1.2, the stabilization techniques can be categorized as (1) optical image stabilization (OIS) and (2) digital image stabilization (DIS). The OIS systems have been designed to reduce apparent motion in image sequences by controlling the optical path sensed by sensors such as gyroscopes or accelerometers.

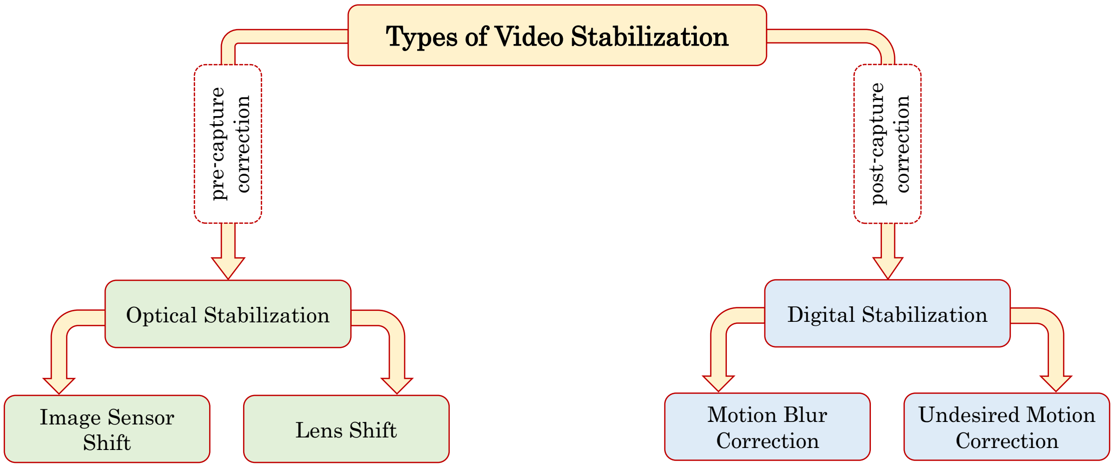
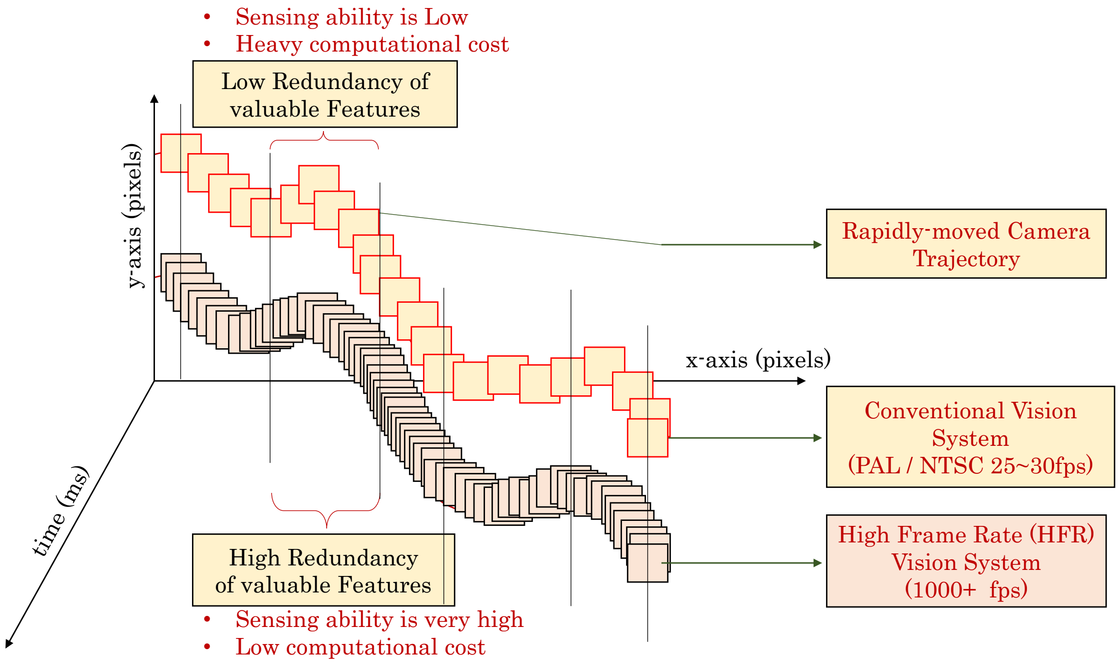


Figure 1.2: Types of Image or Video Stabilization

The lens-shift OIS systems shift their optical path using Opto-mechatronic devices such as a lens-barrel-shift mechanism [26, 27], a fluidic prism [28], a magnetic 3-DOF platform [29], a deformable mirror [30], sensor-shift OIS systems to shift their image sensors using voice coil actuators [31–35], and hand-held OIS systems with multi-DOF gimbal control systems [36–39] have been reported by researchers. The consumer digital cameras have the OIS stabilization functions to remove certain and undesired fluctuating motion while capturing video. These OIS systems can stabilize input images by reducing the motion blur induced by the camera shake. However, conventional systems have difficulty in perfectly reducing large and quick apparent motion by controlling the optical path with sensors that cannot detect any apparent motion in images, due to the physical limitations in the lens-shift or sensor-shift mechanisms. For frame-by-frame image stabilization in a video sequence, the DIS systems can produce a compensated video. The residual fluctuated motion in images can be reduced using various image processing techniques to estimate the local motion vectors, such as block matching [40–43], bit-plane matching [44, 45], Kalman-filter-based prediction [46–50], DFT filtering [51], particle filter [52], scale-invariant feature [53, 54], feature point matching [55–59], and optical flow estimation [60–65]. These systems do not require any additional mechanism or optical device for video stabilization, and they have been used as low-cost video stabilizers in various applications such as airborne shooting [66–72], off-road vehicles [73], and teleoperated applications [74–77], including commercial applications [78–82]. Various approaches have been attempted to achieve real-time DIS systems [83–89] for stabilizing a video sequence with simultaneous video processing at conventional frame rate, whereas, most of them have limited ability to reduce large and quick apparent motion observed in images due to heavy computation in the frame corresponding process.

1.1.2.3 High-speed Vision System as a Sensing Device



• Sensing ability is Low
• Heavy computational cost
Low Redundancy of valuable Features
y-axis (pixels)
x-axis (pixels)
time (ms)
Rapidly-moved Camera Trajectory
Conventional Vision System (PAL / NTSC 25~30fps)
High Frame Rate (HFR) Vision System (1000+ fps)
High Redundancy of valuable Features
• Sensing ability is very high
• Low computational cost

Figure 1.3: An HFR Vision-based Sensing

If a real-time HFR vision system could simultaneously estimate the apparent motion in the images at a high frame rate in a manner like that of conventional sensors, it could be made to function as an HFR displacement and jitter sensor for video mosaicking and stabilization even when the camera or the targeted scene moves quickly, as shown in Figure 1.3. With rapid advancements in computer vision technologies, various real-time HFR vision systems operating at 1000 fps or more have been developed [93], and their effectiveness has been demonstrated in tracking applications such as robot manipulations [94–97], multi-copter tracking [98, 99], camshift tracking [106], multi-object tracking [107], feature point tracking [108], and face tracking [109]. These systems were computationally accelerated by parallel-implementation on field-programmable gate arrays (FPGAs) and graphics processing units (GPUs) to obtain real-time HFR video processing.

Watanabe et al. performed multi-target tracking for 256×256 images at 955 fps using massively parallel coprocessors [100]. Hirai et al. mounted a Hough transform processor on an FPGA [101]. Ishii et al. developed a high-speed vision platform for real-time image processing of 1024×1024 images at 1000 fps [102], and demonstrated its effectiveness via real-time optical flow estimation at 1000 fps [105]. Although these high-speed vision systems can calculate scalar image features in real-time, they cannot be transferred the entire image at high speed to a personal computer (PC) owing to the limitations in the transfer speed of its inner bus. Ishii et al. recently developed IDP Express [103], a high-speed vision platform that can simultaneously process HFR videos using hardware logic, for image processing such as color-histogram-based tracking [106] and multi-object feature extraction [104], which is implemented on an FPGA, and directly maps the HFR videos and processed results onto memory allocated in a PC. Implementing a real-time function to extract feature points in images and match their correspondences between frames on such a high-speed vision platform would make it possible to accelerate image mosaicking at a higher frame-rate than dozens of frames per second, even when the camera moves quickly.

1.1.2.4 Our Proposed Methods

(a) HFR Vision-based Real-time Video Mosaicking:

Concurrent with recent rapid advances in computer technology, real-time image mosaicking systems have also been developed for online visualization of panoramic images at dozens of frames per second. Most of the video cameras have been used in these applications are restricted to standard video signal format and low frame rate; hence, those cameras must be operated slowly without any large displacement bet between the frames. Therefore, there is a demand for the acceleration of computation for image mosaicking and the conducting of real-time image mosaicking at a much higher frame-rate than the dozens of frames per second that is currently the norm.

In this study, we propose a real-time video mosaicking and stabilization systems [108] that use an HFR video sequence for sensing camera motion to compose the wide panoramic image and smooth videos suitable for human eyes. Our proposed method can mosaic 512×512 color images captured at 500 fps as a single synthesized panoramic image in real-time by stitching the images based on their estimated frame-to-frame changes in displacement and orientation.

In the system, feature point extraction is accelerated by implementing a parallel processing circuit module for Harris corner detection, and hundreds of selected feature points in the current frame can be simultaneously corresponded with those in their neighbor ranges in the previous frame, assuming that frame-to-frame image displacement becomes smaller in high-speed vision. The efficacy of our system for improved feature-based real-time image mosaicking at 500 fps was verified by implementing it on a field-programmable gate array (FPGA)-based high-speed vision platform and conducting several experiments: (1) capturing an indoor scene using a camera mounted on a fast-moving two-degrees-of-freedom active vision, (2) capturing an outdoor scene using a hand-held camera that was rapidly moved periodically by hand.

(b) HFR Vision-based Hybrid Video Stabilization:

In this study, we also propose the novel approach of a real-time video stabilization system using an HFR jitter sensing device to realize the computationally efficient technique of digital video stabilization for high-resolution image sequences [114]. This system consists of a high-speed camera to extract and track feature points in gray-level 512×496 image sequences at 1000 fps and a high-resolution Complementary Metal-Oxide Semiconductor (CMOS) camera to capture 2048×2048 image sequences considering their hybridization to achieve real-time stabilization. The high-speed camera functions as a real-time HFR jitter sensing device to measure an apparent jitter movement of the system by considering two ways of computational acceleration; (1) feature point extraction with a parallel processing circuit module of the Harris corner detection and (2) corresponding hundreds of feature points at the current frame to those in the neighbor ranges at the previous frame on the assumption of small frame-to-frame displacement in high-speed vision. The proposed hybrid-camera system can digitally stabilize the 2048×2048 images captured with the high-resolution CMOS camera by compensating the sensed jitter-displacement in real-time for displaying to human eyes on a computer display. The experiments were conducted to demonstrate the effectiveness of hybrid-camera-based digital video stabilization such as (a) verification when the hybrid-camera system in the pan direction in front of a checkered pattern, (b) stabilization in video shooting a photographic pattern when the system moved with a mixed-displacement motions of jitter and constant low-velocity in the pan direction, and (c) stabilization in video shooting a real-world outdoor scene when an operator holding hand-held hybrid-camera module while walking on the stairs.

1.2 Outline of Thesis

This thesis is organized as 6 Chapters including this introduction.

In Chapter 2, the concept of high-speed vision-based video mosaicking and stabilization using a feature point-based approach is proposed, including the fundamentals of global displacement and jitter sensing using an HFR vision system.

In Chapter 3, the algorithms for displacement and jitter sensing are explained along with a description of panoramic image composition and stabilized video composition.

In Chapter 4, the resources used for implementing the proposed method are explained with a detail clarification of the hardware and software components. The multithreaded approach for real-time synthesis is also specified in this chapter.

In Chapter 5, the experiments conducted for verifying the effectiveness of digital video stabilization and video mosaicking are demonstrated including verification and the outdoor demonstration to show the feasibility of the proposed method in a real-world scene as well as discusses the experiments conducted for composing panoramic images at indoor and outdoor scenes. In this chapter, the computational efficiency of our proposed system is also discussed and compared with the various conventional methods for jitter sensing and video stabilization.

Finally, Chapter 6, concludes the contributions of this study and discusses the futuristic improvements to achieve more robust results as well as the real-time implementation of video stabilization and mosaicking in various applications of the real-world.

Chapter 2

Concept of an HFR Vision-based Video Mosaicking and Stabilization

In this chapter, the concept of HFR vision-based video mosaicking and stabilization is explained using novel approaches including the fundamentals of sensing the global displacements and motion anomalies. The high-speed vision-based accelerated platform used for real-time synthesis is also explained in this chapter. As we identified issues associated with the mosaicked panorama as well as in composing smooth video sequences, we proposed various novel methods to overcome conventionally unattended problems. We focus on (a) optimization and acceleration approaches to synthesize a feature point-based displacement and jitter sensing using an HFR vision platform in real-time, (b) an innovative approach on hybrid-dual camera-based image-space resolution compensation for composing high-resolution smooth video sequences, and (c) software multi-threading for simultaneous video composition, simultaneous image acquisition by using hybrid-dual camera module and recording to synthesize in real-time and live display of composed images and videos for operators.

2.1 Optimization and Acceleration

To optimize and accelerate our method for real-time synthesis of video mosaicking and stabilization, we propose, (1) feature-based displacement and jitter sensing using an HFR vision platform, (2) a reduced number of feature points, and (3) a narrowed search range assuming an HFR sequence. The detail explanation is as follows,

2.1.1 Feature-based Displacement and Jitter Sensing using an HFR Vision Platform

In many applications, the image composition at dozens of frames per second is sufficient for the human eyes to monitor a panoramic images and videos, whereas feature tracking at dozens of frames per second is not always stable and accurate when the camera moves rapidly. Most of the feature-based methods used for mosaicking and stabilization are realized by executing (1) feature extraction, (2) feature point matching, (3) frame-by-frame transform estimation, and (4) composition of jitter-compensated image sequences. Corresponding to steps (1)-(3), feature-based motion estimation at the frame rate of conventional cameras is not always stable and there are chances of inaccurate reduction of large apparent motions when a quick movement occurs; leads to heavy computation caused by large image displacements between the frames. In this study, we propose the concept of HFR vision-based displacement and jitter sensing, which is accelerated by an FPGA based vision hardware of our system. As shown in Figure 2.1, the partial implementation of the algorithm i.e., the most computationally expensive feature extraction step on the HFR vision platform, the algorithm is significantly accelerated to achieve in real-time. Gradient-based feature extractors such as the Harris corner detector [110], is suitable for acceleration by hardware logic because the local calculation of brightness gradients can be easily parallelized as pixel-level computation.

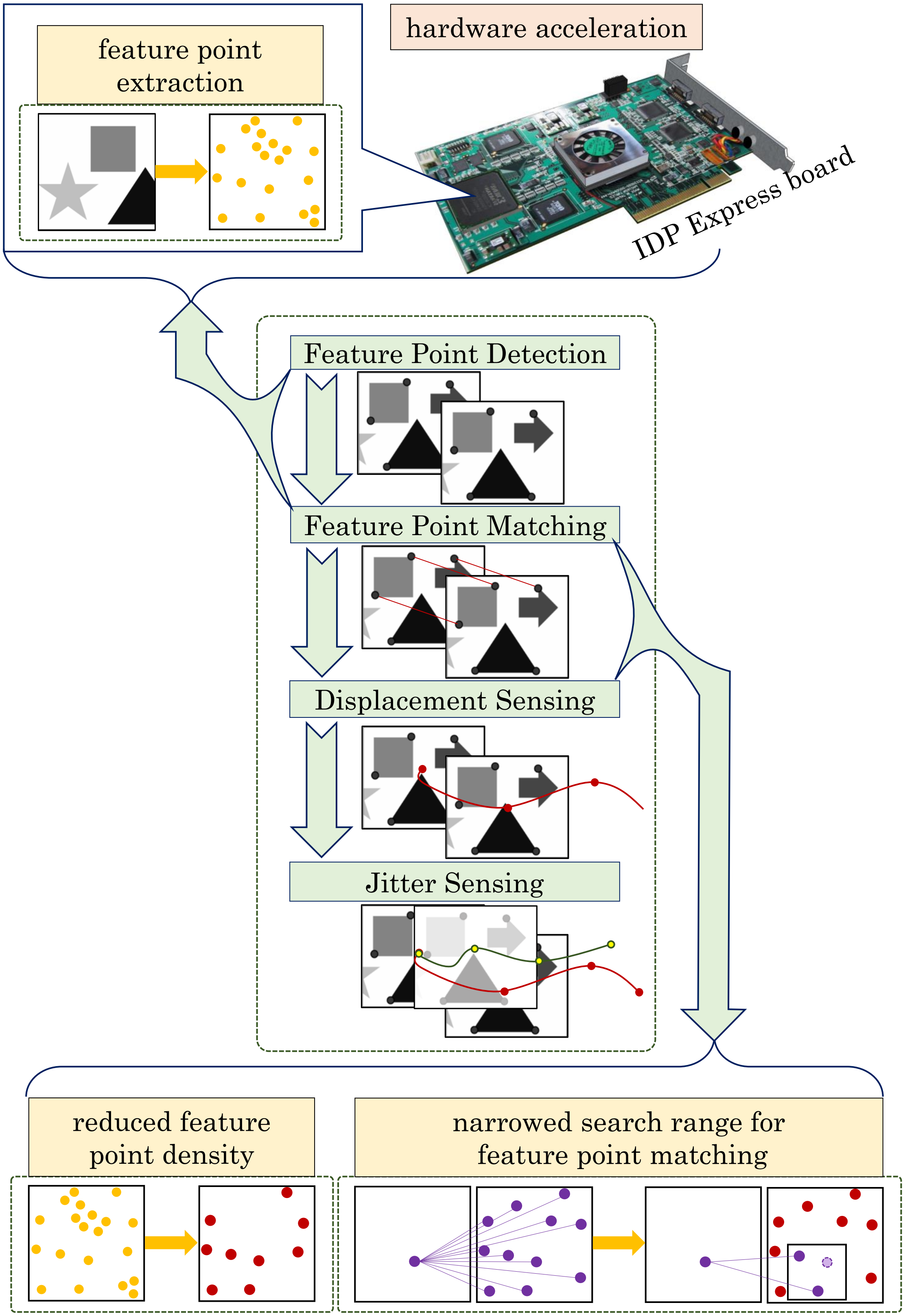


Figure 2.1: Feature-based Displacement and Jitter Sensing using HFR Vision-paltform

2.1.2 A Reduced Number of Feature Points

In the feature point matching for image mosaicking, thousands of feature points are not always necessary for estimating transform parameters between frames. The computational load of feature point matching can be reduced by selecting a smaller number of feature points because the feature-level computation of the order $O(M^2)$ is required in feature point matching, where M is the number of selected feature points. In this study, as shown in Figure 2.1, the number of feature points used in feature point matching is reduced for fast image composition by excluding closely crowded feature points, because they often generate corresponding errors between frames in feature point matching.

2.1.3 A Narrowed Search Range by Assuming HFR Video

Even when a reduced number of feature points are used for the sensing, heavy computation of the order $O(M^2)$ is still required for feature point matching if all the feature points correspond to each other between frames. It can be assumed that frame-to-frame image displacement grows considerably smaller in an HFR video, which allows a smaller search range to be used for points corresponding to points in the consecutive frame. This narrowed search range reduces the inaccuracy and the computational load of feature point matching to corresponding feature points between frames on the order of $O(M)$, even when thousands of feature points are observed from a rapidly moving camera, whereas image composition at a high frame rate incurs a heavy computational load, especially because it requires pixel-level processing. Thus, we introduce a real-time HFR vision-based algorithm that can realize this trade-off between tracking accuracy and computational load in displacement sensing for rapid camera motion, as shown in Figure 2.1.

2.2 Hybrid-dual Camera Module for Image-space Resolution Compensation

Narrowing the search range by assuming temporal redundancy in HFR image sequences can accelerate the process of frame-by-frame motion estimation, whereas video stabilization using HFR image sequences has a shortcoming in image-space resolution and brightness; the former is restricted by the specification of an image sensor as well as the processing power for motion estimation, and the latter depends on its short exposure time, which is less than the frame cycle time of the HFR camera. Thus, we introduce the concept of hybrid-camera-based digital video stabilization that can solve this trade-off between the tracking accuracy in real time motion estimation and the space resolution in composing a compensated video sequence. Generally, real-time video stabilization aims to reduce fluctuated motions in image sequences to generate compensated videos convenient for human eyes on a computer display. Most of the displays are designed to operate at tens of frames per second, which is sufficient for human eyes to perceive it as a smooth movie.

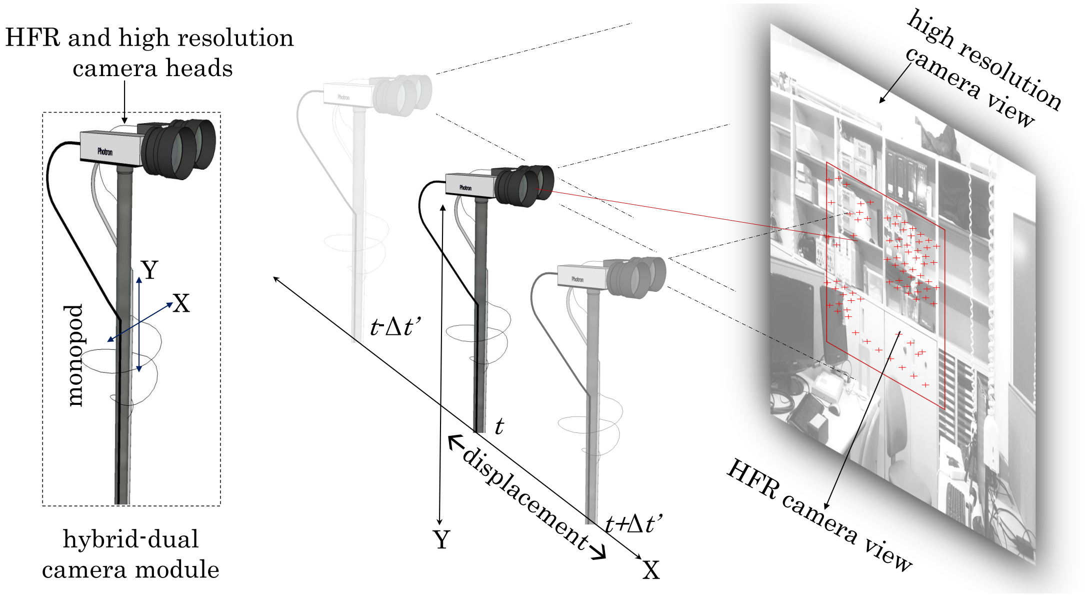


Figure 2.2: Hybrid Approach on Video Stabilization of High-resolution Sequences using an HFR Vision-based Sensing

If a high-resolution camera of $N'_x \times N'_y$ pixels can capture a tens of frames per second video sequence for a view similar to that in the HFR image sequence when mounted on the same platform, both cameras can experience the same desired and undesired motion at the same time. Hence a jitter-compensated $N'_x \times N'_y$ image sequence can be composed in real time without heavy computational complexity for HFR image synthesis; the high-speed vision system works as an HFR jitter sensor to determine jitter-compensation parameters.

The hybrid-dual camera-based system as shown in Figure 2.2, consists of a high-speed vision system that can extract and track the feature points in consecutive images in real-time at thousands of frames for fast apparent motion estimation as an HFR jitter sensor in steps (1)-(3), and a high-resolution camera system for composing compensated high-resolution sequences at dozens of frames per seconds convenient for human eyes in step (4). It is assumed that these camera systems have overlapped views of scenes or objects in the view field.

2.3 Software-based Multi-threading

In our system, various resources are simultaneously executed to handle various tasks such as HFR image acquisition, high-resolution image acquisition, mosaicked panorama and smooth video composition, actuator control, and online data recording to synthesize all the process in real-time. We implemented the proposed system on windows platform, so windows multi-threading and multi-processing application program interfaces (API) are used in our approach.

Chapter 3

Algorithms for Video Mosaicking and Video Stabilization

The steps involved in video mosaicking and video stabilization algorithms are described in this chapter. A feature-based displacement sensing approach is used for both the applications. The mosaicked panorama is composed of stitched images with significant displacement trajectory in the camera, whereas, stabilized video is composed of image sequences of nullified motion anomalies in the camera displacement. Generally, a feature-based displacement sensing is performed using (a) feature point detection, (b) feature point matching, (c) image transformation calculation for sensing frame-by-frame displacement in motion sequences. However, one additional step of (d) undesired motion filtering or compensation is required for sensing jitter in the camera displacement. In both the algorithms used for video mosaicking and stabilization, we used different methods to achieve identical steps to study a suitable way for further optimization. In the case of video mosaicking, we used the Harris Corner detector for extracting feature points from the HFR sequences, Sum of Absolute Difference (SAD) is used for matching feature points and Affine Transformation is used for sensing frame-to-frame displacement in the motion sequences. Whereas, the video stabilization algorithm consists of the same feature detector, Sum of Squared Difference (SSD), Translational Transformation to sense frame-to-frame displacement in the x-y direction and IIR filter to sense jitter in displacement sequences. A detailed explanation of the algorithm is as follows.

3.1 Video Mosaicking

Our algorithm for real-time HFR image mosaicking consists of the following processes.

3.1.1 Feature Detection

(1) Feature point detection

(1-a) Feature extraction

To extract feature points such as upper-left vertexes, we use the following brightness gradient matrix $C(\boldsymbol{x}, t)$:

$$C(\boldsymbol{x}, t_k) = \sum_{\boldsymbol{x} \in N_a(\boldsymbol{x})} \begin{bmatrix} I_x'^2(\boldsymbol{x}, t_k) & I_x'(\boldsymbol{x}, t_k) I_y'(\boldsymbol{x}, t_k) \\ I_x'(\boldsymbol{x}, t_k) I_y'(\boldsymbol{x}, t_k) & I_y'^2(\boldsymbol{x}, t_k) \end{bmatrix}, \tag{3.1}$$

where $N_a(\boldsymbol{x})$ is the $a \times a$ adjacent area of pixel $\boldsymbol{x} = (x, y)$, and $I_x'(\boldsymbol{x}, t)$ and $I_y'(\boldsymbol{x}, t)$ indicate the following positive values of $I_x(\boldsymbol{x}, t)$ and $I_y(\boldsymbol{x}, t)$, respectively:

$$I_\xi'(\boldsymbol{x}, t) = \begin{cases} I_\xi(\boldsymbol{x}, t) & (I_\xi(\boldsymbol{x}, t) > 0) \\ 0 & (\text{otherwise}) \end{cases} \quad (\xi = x, y), \tag{3.2}$$

where $I_x(\boldsymbol{x}, t)$ and $I_y(\boldsymbol{x}, t)$ are the x and y differentials of the input image $I(\boldsymbol{x}, t)$ at pixel $\boldsymbol{x}$ at time t.

$\lambda(\boldsymbol{x}, t)$ is defined as a feature for feature tracking using Harris corner detection [110] as follows:

$$\lambda(\boldsymbol{x}, t) = \det C(\boldsymbol{x}, t) - \kappa(\mathrm{Tr}\, C(\boldsymbol{x}, t))^2. \tag{3.3}$$

Here, κ is a tunable sensitive parameter, and values in the range 0.04–0.15 have been reported as feasible.

(1-b) Feature point detection

Thresholding is conducted for $\lambda(\boldsymbol{x}, t)$ with a threshold λ_T to obtain a map of feature points, $R(\boldsymbol{x}, t)$, as follows:

$$R(\boldsymbol{x}, t) = \begin{cases} 1 & (\lambda(\boldsymbol{x}, t) > \lambda_T) \\ 0 & \text{(otherwise)} \end{cases} \tag{3.4}$$

The number of feature points when $R(\boldsymbol{x}, t) = 1$ is counted in the $p \times p$ adjacent area of $\boldsymbol{x}$ as follows:

$$P(\boldsymbol{x}, t) = \sum_{\boldsymbol{x}' \in N_p(\boldsymbol{x})} R(\boldsymbol{x}', t), \tag{3.5}$$

where $P(\boldsymbol{x}, t)$ indicates the density of the feature points.

(1-c) Selection of feature points

To reduce the number of feature points, closely crowded feature points are excluded by checking the number of feature points in their neighborhood. The reduced set of feature points $R'(t)$ is calculated at time t as follows:

$$R'(t) = \{\boldsymbol{x} \mid P(\boldsymbol{x}, t) \leq P_0\}, \tag{3.6}$$

where P_0 is a threshold used to sparsely select feature points. We assume that the number of feature points is less than M.

3.1.2 Displacement Sensing

3.1.2.1 Feature Matching

(2-a) Template matching

To correspond feature points at the current time t with those from the previous time $t_p(t) = t - n(t)\Delta t$, template matching is conducted for the selected feature points. Δt is the shortest frame interval of a vision system, and $n(t)$ is an integer. For template matching to correspond the i-th feature point at time t_p belonging to $R'(t_p)$, $\boldsymbol{x}_i(t_p)$ $(1 \leq i \leq M)$, to the i'-th feature point at time t belonging to $R'(t)$, $\boldsymbol{x}_{i'}(t)$ $(1 \leq i' \leq M)$, the sum of absolute difference is calculated as follows:

$$E(i', i; t, t_p) = \sum_{\boldsymbol{\xi}=(\xi,\eta)\in W_m} |I(\boldsymbol{x}_{i'}(t)+\boldsymbol{\xi}, t) - I(\boldsymbol{x}_i(t_p)+\boldsymbol{\xi}, t_p)|, \tag{3.7}$$

where W_m is an $m \times m$ template matching window.

To reduce the number of mismatched points, $\hat{\boldsymbol{x}}(\boldsymbol{x}_i(t_p); t)$, which indicates the feature point at time t corresponding to the i-th one, $\boldsymbol{x}_i(t_p)$, at time t_p, and $\hat{\boldsymbol{x}}(\boldsymbol{x}_{i'}(t); t_p)$, which indicates the feature point at time t_p corresponding to the i'-th one, $\boldsymbol{x}_{i'}(t)$, at time t, are bidirectionally searched by selecting the feature points when $E(i', i; t, t_p)$ is at a minimum in their adjacent areas as follows:

$$\hat{\boldsymbol{x}}(\boldsymbol{x}_i(t_p); t) = \boldsymbol{x}_{i'(i)}(t) = \arg\min_{\boldsymbol{x}_{i'}(t)\in N_b(\boldsymbol{x}_i(t_p))} E(i', i; t, t_p), \tag{3.8}$$

$$\hat{\boldsymbol{x}}(\boldsymbol{x}_{i'}(t); t_p) = \boldsymbol{x}_{i(i')}(t_p) = \arg\min_{\boldsymbol{x}_i(t_p)\in N_b(\boldsymbol{x}_{i'}(t))} E(i', i; t, t_p), \tag{3.9}$$

where $i'(i)$ and $i(i')$ are determined as the index numbers of the feature point at time t, corresponding to $\boldsymbol{x}_i(t_p)$, and the feature point at time t_p, corresponding to $\boldsymbol{x}_{i'}(t)$, respectively. The pair of feature points between times t and t_p are selected when the corresponding feature points are mutually selected as the same points as follows:

$$\tilde{\boldsymbol{x}}(\boldsymbol{x}_i(t_p);t) = \begin{cases} \hat{\boldsymbol{x}}(\boldsymbol{x}_i(t_p);t) & (i = i(i'(i))) \\ \emptyset & (\text{otherwise}) \end{cases} \tag{3.10}$$

where $\emptyset$ indicates that no feature point at time t corresponds to the i-th feature point, $\boldsymbol{x}_i(t_p)$, at time t_p, and the feature points at time t that have no corresponding point at time t_p, are adjudged to be newly appearing feature points.

It is assumed that the image displacement between times t and t_p is small, and the feature point $\boldsymbol{x}_i(t)$ at time t can be matched with a feature point at time t_p in the $b \times b$ adjacent area of $\boldsymbol{x}_i(t)$. The processes described in Eqs. (3.8)–(3.10) are conducted for the feature points belonging to $R'(t_p)$ and $R'(t)$, and the computational load of feature point matching can be reduced in the order of $O(M)$ by setting a narrowed search range with a small value for b.

(2-b) Frame interval selection

To avoid accumulated registration errors caused by small frame-to-frame displacements, the previous time $t_p(t + \Delta t)$ for feature point matching at the next time $t + \Delta t$, is adjusted for having a certain image displacement between times $t + \Delta t$ and $t_p(t + \Delta t)$ using the following averaged distance among the selected pairs of feature points at times t and $t_p(t)$:

$$\bar{d}(t;t_p(t)) = \frac{\displaystyle\sum_{\boldsymbol{x}_i(t_p(t)) \in Q(t;t_p(t))} |\tilde{\boldsymbol{x}}(\boldsymbol{x}_i(t_p(t));t) - \boldsymbol{x}_i(t_p(t))|}{S(Q(t;t_p(t)))}, \tag{3.11}$$

where $Q(t;t_p(t))$ is a set of the feature points $\boldsymbol{x}_i(t_p(t))$ at time $t_p(t)$ that satisfy $\tilde{\boldsymbol{x}}(\boldsymbol{x}_i(t_p(t));t) \neq \emptyset$ $(i = 1, \cdots, M)$, that is, their corresponding points exist at time t. $S(Q(t;t_p(t)))$ is the number of elements belonging to $Q(t;t_p(t))$.

Depending on whether $\bar{d}(t; t_p(t))$ is larger than d_0, $n(t+\Delta t)$ is determined for feature point matching at time $t + \Delta t$; the frame interval of feature point matching is initially set to Δt.

(2-b-i) $\bar{d}(t; t_p(t)) \leq d_0$,

The image displacement between times t and $t_p(t)$ is too small for feature point matching. To increase the displacement at the next point in time, $t + \Delta t$, the parameter $n(t+\Delta t)$ is incremented as $n(t+\Delta t) = n(t)+1$; the feature points at time $t+\Delta t$ are matched with those at time $t_p(t + \Delta t) = t - n(t)\Delta t$.

Without conducting any process in steps (3) and (4) below, return to step (1) for the next time, $t + \Delta t$.

(2-b-ii) $\bar{d}(t; t_p(t)) > d_0$

To reset the frame-to-frame displacement at time $t + \Delta t$, $n(t + \Delta t)$ is set to one; the feature points at time $t + \Delta t$ are matched with those at time $t_p(t + \Delta t) = t$.

Thereafter, go to steps (3) and (4); the selected pairs of feature points, $\boldsymbol{x}_i(t)$ ($\in Q(t; t_p(t))$) and $\tilde{\boldsymbol{x}}(\boldsymbol{x}_i(t); t_p(t))$, are used in steps (3) and (4).

3.1.2.2 Affine Transformation

Affine parameters between the two images at times t and t_p are estimated using the selected pairs of feature points, $\boldsymbol{x}_i(t)$ and $\tilde{\boldsymbol{x}}(\boldsymbol{x}_i(t), t_p)$, as follows:

$$\begin{aligned} \boldsymbol{x}_i^T(t) &= \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \tilde{\boldsymbol{x}}^T(\boldsymbol{x}_i(t); t_p) + \begin{bmatrix} a_5 \\ a_6 \end{bmatrix} && (3.12) \\ &= \boldsymbol{A}(t; t_p)\tilde{\boldsymbol{x}}^T(\boldsymbol{x}_i(t); t_p) + \boldsymbol{b}^T(t; t_p), && (3.13) \end{aligned}$$

where a_j $(j = 1, \cdots, 6)$ are components of the matrix $\boldsymbol{A}(t; t_p)$ and vector $\boldsymbol{b}(t; t_p)$ that express the two-dimensional affine transform relationship between the two images at times t and t_p. They are estimated by minimizing the following Tukey's biweight evaluation functions, E_1 and E_2:

$$E_l = \sum_{\boldsymbol{x}_i(t_p) \in Q(t;t_p)} w_{li} \cdot d_{li}^2 \quad (l = 1, 2) \tag{3.14}$$

Deviations d_{1i} and d_{2i} are given as follows:

$$d_{1i} = |x''_i - (a_1 x'_i + a_2 y'_i + a_5)| \tag{3.15}$$

$$d_{2i} = |y''_i - (a_3 x'_i + a_4 y'_i + a_6)|, \tag{3.16}$$

where $\tilde{\boldsymbol{x}}(\boldsymbol{x}_i(t); t_p) = (x'_i, y'_i)$ and $\boldsymbol{x}''_i(t) = (x''_i, y''_i)$ indicate the temporally estimated location of the i-th feature point at time t in Tukey's biweight method. The following processes are iteratively executed to reduce estimation errors caused by distantly mismatched pairs of feature points. Initially, the weights w_{1i} and w_{2i} are set to one and $\boldsymbol{x}''_i(t)$ set to $\boldsymbol{x}_i(t)$.

(3-a) Affine parameter estimation

Using the least squares method to minimize E_1 and E_2, affine parameters a_j $(j = 1, \cdots, 6)$ are estimated by calculating the weighted product sums of the xy coordinates of the selected feature points as follows:

$$\begin{bmatrix} a_1 \\ a_2 \\ a_5 \end{bmatrix} = \begin{bmatrix} \sum_i w_{1i} x_i'^2 & \sum_i w_{1i} x_i' y_i' & \sum_i w_{1i} x_i' \\ \sum_i w_{1i} x_i' y_i' & \sum_i w_{1i} y_i'^2 & \sum_i w_{1i} y_i' \\ \sum_i w_{1i} x_i' & \sum_i w_{1i} y_i' & \sum_i w_{1i} \end{bmatrix}^{-1} \begin{bmatrix} \sum_i w_{1i} x_i'' x_i' \\ \sum_i w_{1i} x_i'' y_i' \\ \sum_i w_{1i} x_i'' \end{bmatrix} \tag{3.17}$$

$$\begin{bmatrix} a_3 \\ a_4 \\ a_6 \end{bmatrix} = \begin{bmatrix} \sum_i w_{2i} x_i'^2 & \sum_i w_{2i} x_i' y_i' & \sum_i w_{2i} x_i' \\ \sum_i w_{2i} x_i' y_i' & \sum_i w_{2i} y_i'^2 & \sum_i w_{2i} y_i' \\ \sum_i w_{2i} x_i' & \sum_i w_{2i} y_i' & \sum_i w_{2i} \end{bmatrix}^{-1} \begin{bmatrix} \sum_i w_{2i} y_i'' x_i' \\ \sum_i w_{2i} y_i'' y_i' \\ \sum_i w_{2i} y_i'' \end{bmatrix} \tag{3.18}$$

Using the affine parameters estimated in Eqs. (3.17) and (3.18), the temporally estimated locations of feature points can be updated as follows:

$$\boldsymbol{x}_i''^T(t) = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \tilde{\boldsymbol{x}}^T(\boldsymbol{x}_i(t); t_p) + \begin{bmatrix} a_5 \\ a_6 \end{bmatrix}, \tag{3.19}$$

(3-b) Updating weights

To reduce the errors caused by distantly mismatched pairs of feature points, w_{1i} and w_{2i} are updated for the i-th pair of feature points using deviations d_{1i} and d_{2i} as follows:

$$\begin{bmatrix} w_{1i} \\ w_{2i} \end{bmatrix} = \begin{cases} \begin{bmatrix} \left(1 - \dfrac{d_{1i}^2}{W_u^2}\right)^2 \\ \left(1 - \dfrac{d_{2i}^2}{W_u^2}\right)^2 \end{bmatrix} & (d_{1i} \leq W_u, d_{2i} \leq W_u) \\ \begin{bmatrix} 0 \\ 0 \end{bmatrix} & \text{(otherwise)} \end{cases}, \tag{3.20}$$

W_u is a parameter used to determine the cut point for evaluation functions at u time iteration. As the number of iterations, u, is larger, W_u is set to a smaller value. Increment u by one and return to step (3-a) during $u \leq U$.

After U time iterations, the affine parameters between the two images at times t and t_p are determined. The affine transform matrix and vector, $\boldsymbol{A}(t; t_p)$ and $\boldsymbol{b}(t; t_p)$, are accumulated with the affine parameters estimated at time t_p as follows:

$$\begin{aligned} \boldsymbol{A}(t) &= \boldsymbol{A}(t; t_p)\boldsymbol{A}(t_p) && (3.21) \\ \boldsymbol{b}^T(t) &= \boldsymbol{A}(t_p)\boldsymbol{b}^T(t; t_p) + \boldsymbol{b}^T(t_p) && (3.22) \end{aligned}$$

where $\boldsymbol{A}(t)$ and $\boldsymbol{b}(t)$ indicate the affine parameters of the input image at time t, compared with that at time 0; $\boldsymbol{A}(0)$ and $\boldsymbol{b}(0)$ are given as the unit matrix and zero vector, respectively.

3.1.3 Mosaicked Panorama Composition

To monitor the panoramic image at intervals of $\Delta t'$, the panoramic image $G(\boldsymbol{x}, t)$ is updated by attaching an affine-transformed input image at time t to the panoramic image $G(\boldsymbol{x}, t - \Delta t')$ at time $t - \Delta t'$ as follows:

$$G(\boldsymbol{x}, t) = \begin{cases} I(\boldsymbol{x}', t) & (\text{if } I(\boldsymbol{x}', t) \neq \emptyset) \\ G(\boldsymbol{x}, t - \Delta t') & (\text{otherwise}), \end{cases} \qquad (3.23)$$

where $G(\boldsymbol{x}, 0)$ is initially given as an empty image; $\boldsymbol{x}'$ is the affine-transformed coordinate vector at time t as follows:

$$\boldsymbol{x}'^{T} = \boldsymbol{A}(t)\boldsymbol{x}^{T} + \boldsymbol{b}^{T}(t), \tag{3.24}$$

Our algorithm can be efficiently executed as a multi-rate video processing method. The interval in steps (1) and (2) can be set to the shortest frame interval Δt for accurate feature point tracking in an HFR video sequence, while that in step (3) depends on the frame-to-frame image displacement; it is the shortest frame interval when there is significant camera motion, and it becomes larger when the camera motion becomes smaller. On the other hand, the interval in step (4), $\Delta t'$, can be independently set to dozens of milliseconds to enable the human eye to monitor the panoramic image.

3.2 Video Stabilization

Our algorithm for hybrid-camera-based digital video stabilization consists of the following processes. In the steps of (a) feature point extraction, we used the same algorithms as those used in real-time image mosaicking using an HFR video [108], considering the implementation of parallelized gradient-based feature extraction on an FPGA-based high-speed vision platform.

3.2.1 Feature Detection

The Harris corner feature [110], $\lambda(\boldsymbol{x}, t_k) = \det C(\boldsymbol{x}, t_k) - \kappa(\mathrm{Tr}\ C(\boldsymbol{x}, t_k))^2$ at time t_k, is computed using the following gradient matrix:

$$C(\boldsymbol{x}, t_k) = \sum_{\boldsymbol{x} \in N_a(\boldsymbol{x})} \begin{bmatrix} I_x'^2(\boldsymbol{x}, t_k) & I_x'(\boldsymbol{x}, t_k)I_y'(\boldsymbol{x}, t_k) \\ I_x'(\boldsymbol{x}, t_k)I_y'(\boldsymbol{x}, t_k) & I_y'^2(\boldsymbol{x}, t_k) \end{bmatrix}, \tag{3.25}$$

where $N_a(\boldsymbol{x})$ is the $a \times a$ adjacent area of pixel $\boldsymbol{x} = (x, y)$. $t_k = k\Delta t$ indicates when the input image $I(\boldsymbol{x}, t)$ at frame k is captured by a high-speed vision system operating at a frame cycle time of Δt. $I'_x(\boldsymbol{x},t)$ and $I'_y(\boldsymbol{x},t)$ indicate the positive values of x and y differentials of the input image $I(\boldsymbol{x},t)$ at pixel $\boldsymbol{x}$ at time t, $I_x(\boldsymbol{x},t)$ and $I_y(\boldsymbol{x},t)$, respectively. κ is a tunable sensitive parameter, and values in the range 0.04–0.15 have been reported as feasible.

The number of feature points in the $p \times p$ adjacent area of $\boldsymbol{x}$ is computed as the density of feature points by thresholding $\lambda(\boldsymbol{x},t_k)$ with a threshold λ_T as follows:

$$P(\boldsymbol{x},t_k) = \sum_{\boldsymbol{x}' \in N_p(\boldsymbol{x})} R(\boldsymbol{x}',t_k), \qquad R(\boldsymbol{x},t_k) = \begin{cases} 1 & (\lambda(\boldsymbol{x},t_k) > \lambda_T) \\ 0 & (\text{otherwise}) \end{cases} \tag{3.26}$$

where $R(\boldsymbol{x},t)$ is a map of feature points.

Closely crowded feature points are excluded by counting the number of feature points in the neighborhood. The reduced set of feature points is calculated as $R'(t_k) = \{\boldsymbol{x} \mid P(\boldsymbol{x},t_k) \leq P_0\}$ by thresholding $P(t_k)$ with a threshold P_0. It is assumed that the number of feature points is less than M.

3.2.2 Jitter Sensing

3.2.2.1 Feature Matching

To enable correspondence between feature points at the current time t_k and those at the previous time $t_{k-1} = (k-1)\Delta t$, template matching is conducted for all the selected feature points in an image.

To enable the correspondence of the i-th feature point at time t_{k-1} belonging to $R'(t_{k-1})$, $\boldsymbol{x}_i(t_{k-1})$ $(1 \leq i \leq M)$, to the i'-th feature point at time t_k belonging to $R'(t_k)$, $\boldsymbol{x}_{i'}(t_k)$ $(1 \leq i' \leq M)$, the sum of squared differences is calculated in the window W_m of $m \times m$ pixels as follows:

$$E(i', i; t, t_{k-1}) = \sum_{\xi=(\xi,\eta)\in W_m} \|I(\boldsymbol{x}_{i'}(t) + \xi, t) - I(\boldsymbol{x}_i(t_{k-1}) + \xi, t_{k-1})\|^2. \tag{3.27}$$

To decrease the number of mismatched points, $\hat{\boldsymbol{x}}(\boldsymbol{x}_i(t_{k-1}); t_k)$ and $\hat{\boldsymbol{x}}(\boldsymbol{x}_{i'}(t); t_{k-1})$, which indicate the feature point at time t_k corresponding to the i-th feature point $\boldsymbol{x}_i(t_{k-1})$ at time t_{k-1}, and the feature point at time t_{k-1} corresponding to the i'-th feature point $\boldsymbol{x}_{i'}(t_k)$ at time t_k, respectively, are bidirectionally searched so that $E(i', i; t_k, t_{k-1})$ is minimal in their adjacent areas as follows:

$$\hat{\boldsymbol{x}}(\boldsymbol{x}_i(t_{k-1}); t_k) = \boldsymbol{x}_{i'(i)}(t_k) = \underset{\boldsymbol{x}_{i'}(t_k)\in N_b(\boldsymbol{x}_i(t_{k-1}))}{\arg\min} E(i', i; t_k, t_{k-1}), \tag{3.28}$$

$$\hat{\boldsymbol{x}}(\boldsymbol{x}_{i'}(t_k); t_{k-1}) = \boldsymbol{x}_{i(i')}(t_{k-1}) = \underset{\boldsymbol{x}_i(t_{k-1})\in N_b(\boldsymbol{x}_{i'}(t_k))}{\arg\min} E(i', i; t_k, t_{k-1}), \tag{3.29}$$

where $i'(i)$ and $i(i')$ are the index numbers of the feature point at time t_k corresponding to $\boldsymbol{x}_i(t_{k-1})$, and that at time t_{k-1} corresponding to $\boldsymbol{x}_{i'}(t_k)$, respectively. According to mutual selection of the corresponding feature points, the pair of feature points between time t_k and t_{k-1} are selected as follows:

$$\tilde{\boldsymbol{x}}_i(t_k) = \begin{cases} \hat{\boldsymbol{x}}(\boldsymbol{x}_i(t_{k-1}); t_k) & (i = i(i'(i))) \\ \emptyset & (\text{otherwise}) \end{cases}, \tag{3.30}$$

$$f_i(t_k) = \begin{cases} 1 & (i = i(i'(i))) \\ 0 & (\text{otherwise}) \end{cases}, \tag{3.31}$$

where $f_i(t_k)$ indicates whether there are feature points at time t_k or not, corresponding to the i-th feature point $\boldsymbol{x}_i(t_{k-1})$ at time t_{k-1}.

On the assumption that the frame-by-frame image-displacement between time t_k

and t_{k-1} is small, the feature point $\boldsymbol{x}_i(t_k)$ at time t_k is matched with a feature point at time t_{k-1} in the $b \times b$ adjacent area of $\boldsymbol{x}_i(t_k)$; the computational load of feature point matching is reduced in the order of $O(M)$ by setting a narrowed search range. For all the feature points belonging to $R'(t_{k-1})$ and $R'(t_k)$, the processes described in Eq. (3.28)–(3.31) are conducted, and $M'(t_k)(\leq M)$ pairs of feature points are selected for jitter sensing, where $M'(t_k) = \sum_{i=1}^{M} f_i(t_k)$.

3.2.2.2 Translational Transformation

Assuming that the image-displacement between time t_k and t_{k-1} is translational motion, the velocity $\boldsymbol{v}(t_k)$ at time t_k is estimated by averaging the positions of selected pairs of feature points as follows:

$$\boldsymbol{v}(t_k) = \frac{1}{\Delta t} \cdot \frac{1}{M'(t_k)} \sum_{i=1}^{M} f_i(t_k)(\tilde{\boldsymbol{x}}_i(t_k) - \boldsymbol{x}_i(t_{k-1})), \tag{3.32}$$

Jitter displacement $\boldsymbol{d}(t_k)$ is computed at time t_k by accumulating the estimated velocity $\boldsymbol{v}(t_k)$ as follows:

$$\boldsymbol{d}(t_k) = \boldsymbol{d}(t_{k-1}) + \boldsymbol{v}(v_{k-1}) \cdot \Delta t, \tag{3.33}$$

where the displacement at time $t = t_0 = 0$ is initially set to $\boldsymbol{d}(t_0) = \boldsymbol{d}(0) = \boldsymbol{0}$.

3.2.2.3 Motion Anomalies Filtering

The high-frequency component of jitter displacement $\boldsymbol{d}_{cut}(t_k)$, which is the camera jitter movement intended for removal is extracted using the following high-pass IIR filter,

$$\boldsymbol{d}_{cut}(t_k) = \mathrm{IIR}(\boldsymbol{d}_k, \boldsymbol{d}_{k-1}, \cdots, \boldsymbol{d}_{k-D}; f_{cut}), \tag{3.34}$$

where the order of the IIR filter is D;

it is designed to exclude the low-frequency component of velocity lower than a cut-off frequency f_{cut}.

3.2.3 High-resolution Stabilized-video Composition

When the high-resolution input image $I'(\boldsymbol{x}', t'_{k'})$ at frame k' is captured at time $t'_{k'} = k'\Delta t'$ by a high-resolution camera operating at a frame cycle time of $\Delta t'$, which is much larger than that of the high-speed vision system, Δt, the stabilized high-resolution image $S(\boldsymbol{x}', t'_{k'})$ is composed by displacing $I'(\boldsymbol{x}', t'_{k'})$ with the high-frequency component of jitter displacement $\boldsymbol{d}_{cut}(\hat{t}'_{k'})$ as follows:

$$S(\boldsymbol{x}', t'_{k'}) = I'(\boldsymbol{x}' - l \cdot \boldsymbol{d}_{cut}(\hat{t}'_{k'}), t'_{k'}), \tag{3.35}$$

where $\boldsymbol{x}' = l\boldsymbol{x}$ indicates the image coordinate system of the high-resolution camera; its resolution is l times that of the high-speed vision system. $\hat{t}'_{k'}$ is the time when the high-speed vision system captures its image at the nearest frame after time $t'_{k'}$ when the high-resolution camera captures its image as follows:

$$\hat{t}'_{k'} = \left\lceil \frac{t'_{k'}}{\Delta t} \right\rceil \Delta t, \tag{3.36}$$

where $\lceil a \rceil$ indicates the minimum integer, which is larger than a.

In this way, video stabilization of high-resolution image sequences can be achieved in real time by image composition using input sequences based on a high-frequency-displacement component sensed by executing the high-speed vision system as an HFR jitter sensor.

Chapter 4

System Implementation

In this chapter, our system is described with details specifications. Since we used a feature-based approach to implement the video mosaicking and stabilization, there are identical resources used in both the applications. Resources such as an HFR camera, a frame grabbing board, a frame grabbing hardware logic and a feature detection hardware logic are the common resources for both the applications. The details explanation on each resource are as follows,

4.1 IDP Express HFR Vision System

To realize real-time video mosaicking and stabilization at a high frame rate, we implemented our algorithm on the FPGA-based high-speed vision platform, the IDP Express [103] (IDP-e), which consists of a compact camera head, a dedicated FPGA board (IDP-e board), and a personal computer (PC). Figure 4.1 shows the general configuration of IDP-e, whereas Figure 4.2 gives an overview of the camera head and the IDP-e board.

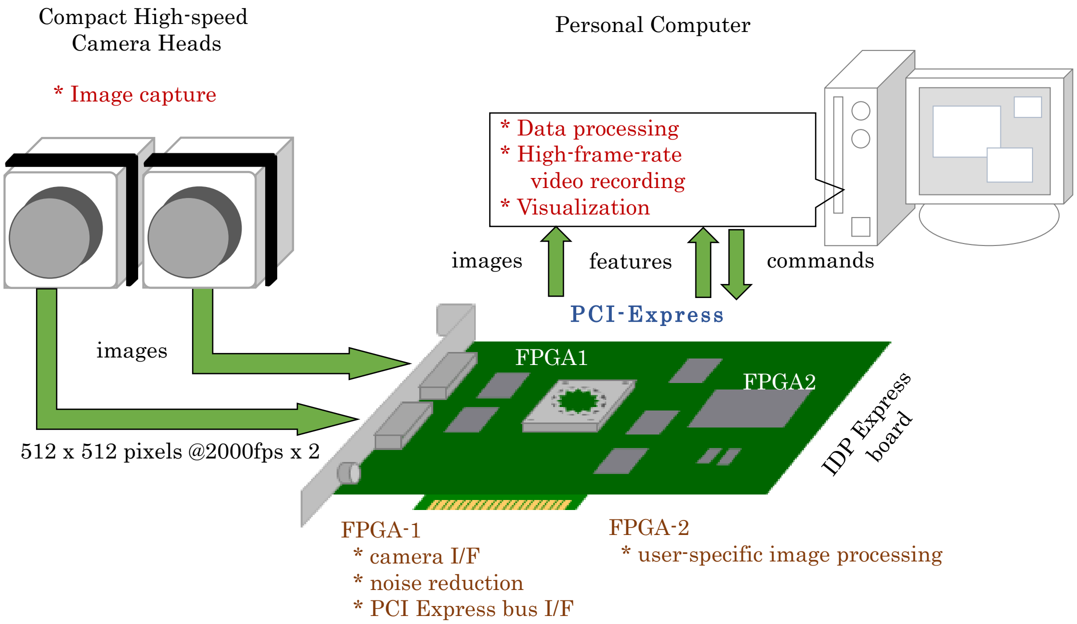
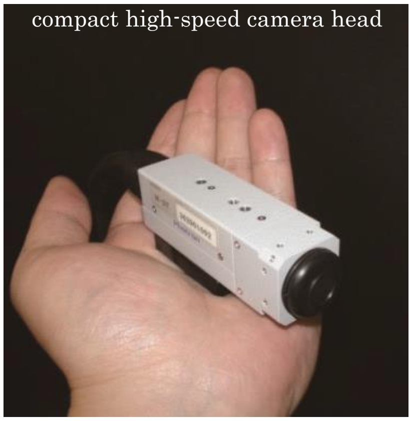


Figure 4.1: Configuration of IDP-e

The camera head is compactly designed; its dimensions and weight are 23×23×77 mm and 145 g, respectively, when no lens is mounted. It can easily be held by a human hand and can also easily be mounted on movable objects for image mosaicking with a fast-moving camera. The housing of the camera head is designed to facilitate the attachment of a compact NF-mount lens.



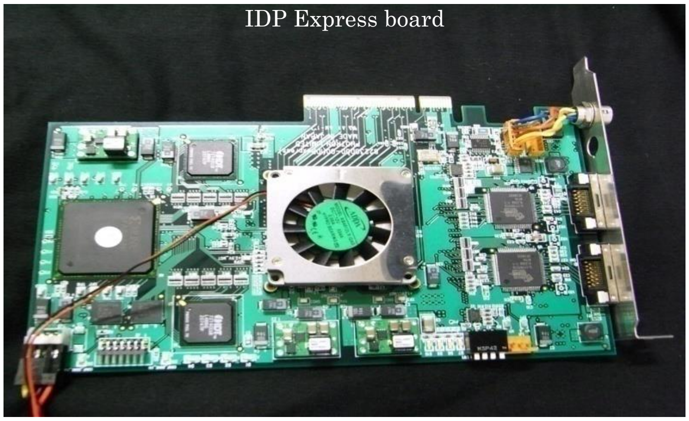


Figure 4.2: Photos of IDP-e (left) Camera Head and (right) Capture Board

It can also be extended to use a C-mount lens via a lens attachment. On the camera head, 8-bit Mono and RGB images of 512×512 pixels can be captured at 2000 fps with a Bayer color filter on its CMOS image sensor of 512×512 pixels; the sensor and pixel sizes are 5.12 mm×5.12 mm and 10 μm×10 μm, respectively.

4.1.1 IDP-e Functional Blocks

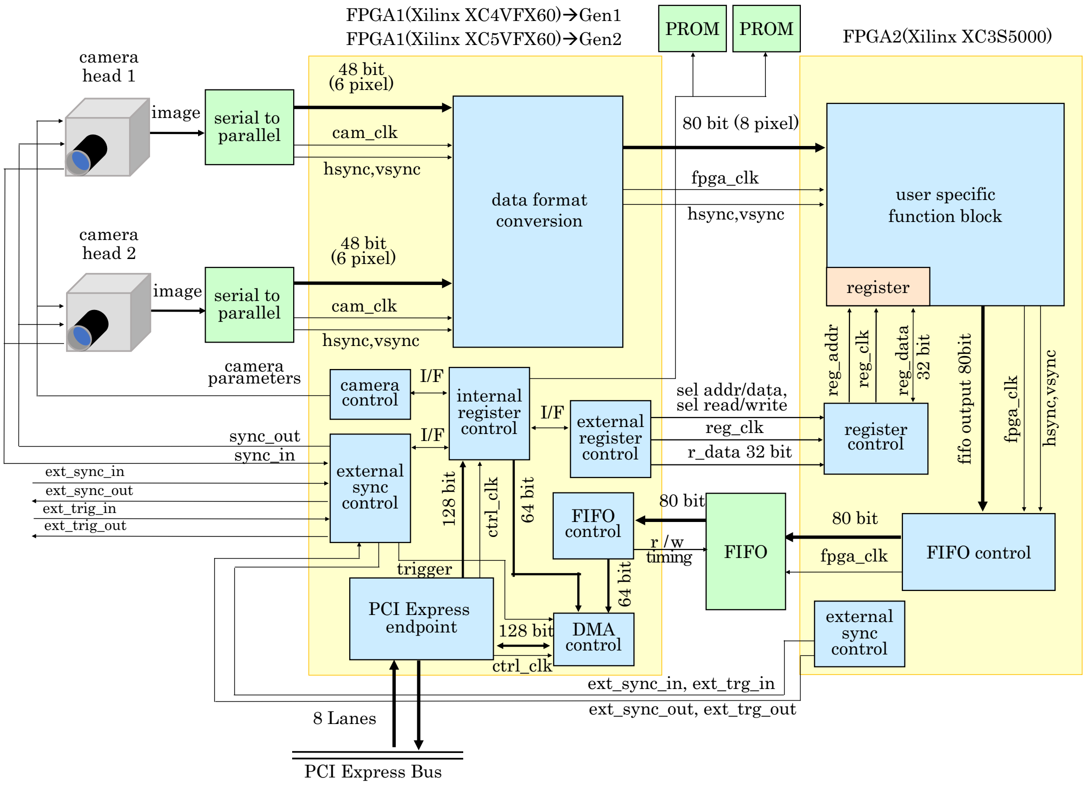


Figure 4.3: Functional Block Diagram of the IDP-e High-speed Vision Board

The IDP-e board was designed for high-speed processing and recording of 512×512 color images transferred at 2000 fps. Figure 4.3 is a function block diagram of the IDP-e board. The board has two camera inputs, a Virtex series FPGA (Xilinx XCVFX60) for a camera I/O and PCI-e bus controls, a user-specified Spartan-3 FPGA (Xilinx XC3S5000-4FG900) for hardware implementation of the algorithms by users, and several peripheral I/O circuits. Details on the image data processing path and transfer in the IDP Express board were given in [103]. In this study, 512×512 images from a single camera head (camera head one) are processed for both the applications. The 512×512 input images, and the processed results on the IDP-e board, can be memory-mapped via 16-lane PCI-e buses in real-time at 2000 fps onto the allocated memories in the PC.

The hardware logic implementation on user-specified FPGA has two variants of modules for both the applications. In the case of video mosaicking, a color camera is used to capture the HFR sequences, as well as another camera is also interfaced with the IDP-e board to manage the bandwidth for mapping color images and the extracted features to the PC memory simultaneously. However, a mono HFR camera is interfaced with IDP-e for the stabilization application. We also optimized our hardware logic to utilize single-camera bandwidth by appending extracted features at the bottom of the HFR sequences.

4.2 HFR Vision-based Video Mosaicking

4.2.1 Hardware Implementation on Spartan-3 FPGA

We designed a circuit module for feature point detection in the user-specified FPGA on the IDP-e board for the acceleration of steps feature detection of our algorithms; they have a computational complexity of $O(M^2)$. Whereas, the rest of the steps, which have a computational complexity of $O(M)$, were software-implemented on the PC because they are accelerated by reducing the number of feature points required for feature point matching. Figure 4.4 depicts the schematic data flow of the feature point extraction circuit module implemented in the user-specified FPGA. The timing chart for the control signals is described in Figure 4.5. The circuit module consists of a color-to-gray converter, a Harris feature extractor, a feature point counter, and a data selector for FIFO output. Raw 8-bit 512×512 input color images with a Bayer color filter, $F(\boldsymbol{x},t)$, are scanned in units of four pixels from the upper left to the lower right using X and Y address signals at 151.2 MHz. The color-to-gray converter can convert RGB images into 8-bit gray-level images, $I(\boldsymbol{x},t)$, in parallel with the four pixels after RGB conversion for input images with a Bayer color filter.

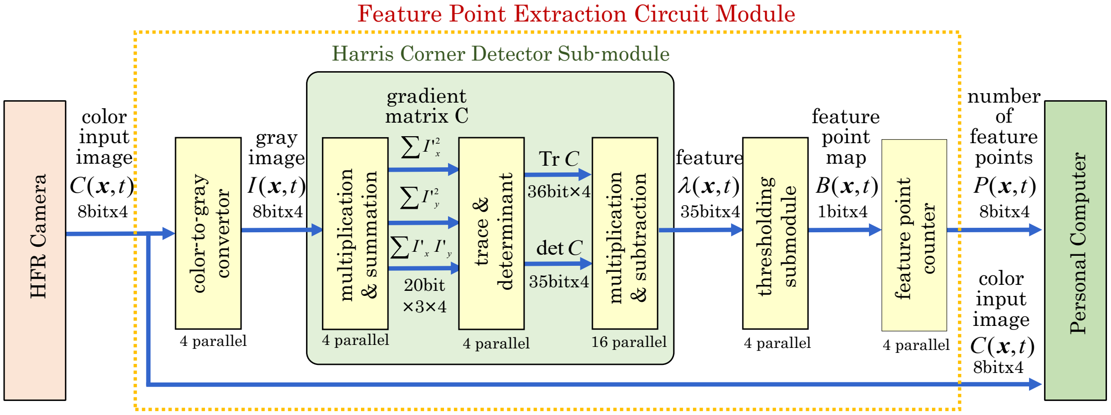

Feature Point Extraction Circuit Module
Harris Corner Detector Sub-module
HFR Camera
color input image $C(x,t)$ 8bitx4
color-to-gray convertor
4 parallel
gray image $I(x,t)$ 8bitx4
multiplication & summation
4 parallel
gradient matrix C
$\sum I'^2_x$
$\sum I'^2_y$
$\sum I'_x I'_y$
20bit ×3×4
trace & determinant
4 parallel
Tr C
36bit×4
det C
35bitx4
multiplication & subtraction
16 parallel
feature $\lambda(x,t)$ 35bitx4
thresholding submodule
4 parallel
feature point map $B(x,t)$ 1bitx4
feature point counter
4 parallel
number of feature points $P(x,t)$ 8bitx4
color input image $C(x,t)$ 8bitx4
Personal Computer

Figure 4.4: Schematic Data Flow of Feature Point Extraction Circuit Module

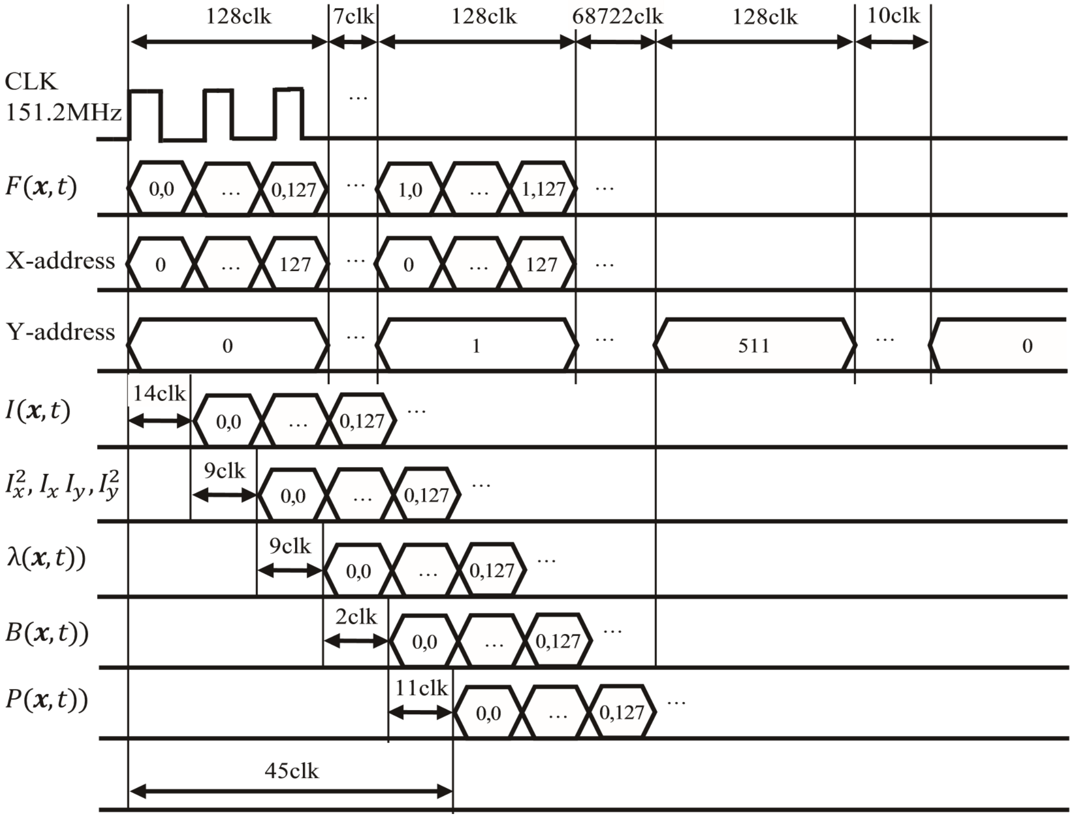


Figure 4.5: Timing Chart for Control Signal

The Harris feature detector can calculate I'_x and I'_y, which are the positive values of partial derivatives I_x and I_y, using 3×3 Prewitt operators, and then calculate their product sums $\sum I'^2_x$, $\sum I'_x I'_y$, and $\sum I'^2_y$ in the adjacent 3×3 pixels ($a = 3$) as 20-bit data using

160 adders and 12 multipliers in parallel in units of four pixels with a 151.2 MHz clock. For the convenience of hardware implementation, the feature for Harris corner detection, $\lambda(\boldsymbol{x}, t)$, is outputted as 35-bit data by subtracting det $C(\boldsymbol{x}, t)$ with a 4-bit shift value of (Tr $C(\boldsymbol{x}, t))^2$ in parallel in units of four pixels when $\kappa = 0.0625$; 16 multipliers, 24 adders, and four 3-bit shifters are implemented for calculating $\lambda(\boldsymbol{x}, t)$.

The feature point counter can obtain the positions of feature points as a 512×512 binary map $B(\boldsymbol{x}, t)$ by thresholding the feature $\lambda(\boldsymbol{x}, t)$ with a threshold λ_T in parallel in units of four pixels at 151.2 MHz; $B(\boldsymbol{x}, t)$ indicates whether a feature point is located at pixel $\boldsymbol{x}$ at time t. The number of feature points in the 5×5 adjacent area ($p = 5$) is counted for the feature points as a 512×512 map $P(\boldsymbol{x}, t)$ using 96 adders in parallel in units of four pixels; $P(\boldsymbol{x}, t)$ is used for checking closely crowded feature points in step (1-c).

In the data selector, the input image $F(\boldsymbol{x}, t)$ and the number of feature points $P(\boldsymbol{x}, t)$ are outputted to FIFO memory for an external PC in parallel in units of four pixels with X and Y address signals at 151.2 MHz. The delay time in outputting $F(\boldsymbol{x}, t)$ and $P(\boldsymbol{x}, t)$ using the feature point extraction circuit module is 46 clocks (1 clock = 13.2 ns) after raster scanning all pixels in a 512×512 input image; 14 clocks for color-to-gray converter; 20 clocks for Harris feature extractor; 11 clocks for feature point counter; and one clock for data selector.

Device Type	Xilinx XC3S5000
Slice	5,864/33,280 (17%)
Slice Flip Flop	9,015/66,560 (13%)
4 input LUT	6,038/66,560 (9%)
Bounded IOB	195/633 (30%)
Block RAM	15/104 (14%)
MULT18X18s	28/104 (26%)
GCLK	2/8 (25%)

Table 4.1: FPGA resource consumption.

Table 4.1 summarizes the resources consumed by the user-specified FPGA (Xilinx XC3S5000-4FG900) in implementing the feature point extraction circuit module.

4.2.2 Software Implementation (Multi-threading) for Video Mosaicked Panorama

Various API functions associated with board control are memory-mapped into the allocated memories on the PC and have been prepared for a 32-bit Windows 7 OS middleware to develop application programs. With these API functions, only the required pixels or processed results can be accessed in real-time. Arbitrary image processing can be programmed for the execution of software on the PC with specifications as an ASUSTek P5E motherboard, Core 2 Quad9300 bulk 2.50 GHz CPU, 4 GB RAM, and a 32-bit Windows 7 OS.

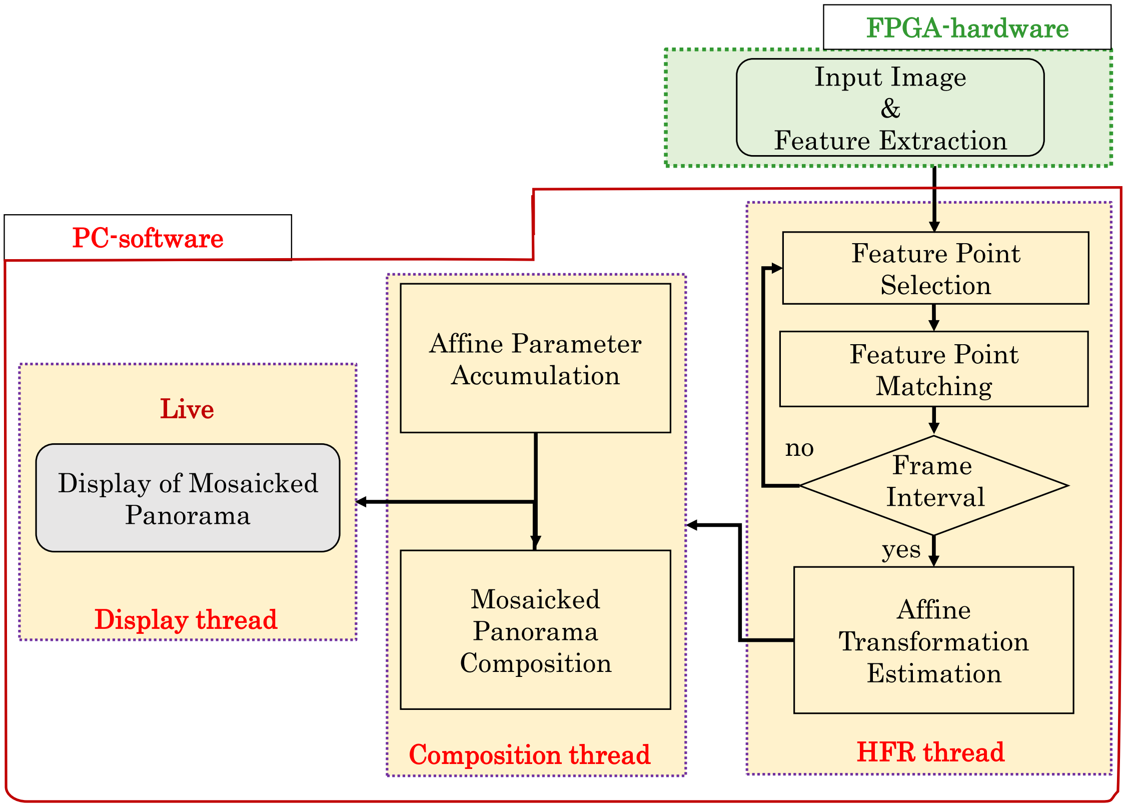

FPGA-hardware
Input Image & Feature Extraction
PC-software
Feature Point Selection
Feature Point Matching
no
Frame Interval
yes
Affine Transformation Estimation
HFR thread
Affine Parameter Accumulation
Mosaicked Panorama Composition
Composition thread
Live
Display of Mosaicked Panorama
Display thread

Figure 4.6: Multi-threaded Approach for Video Mosaicking

By implementing the circuit module described above in the user-specified FPGA, feature detection is accelerated such that the extracted feature points could be outputted to the PC for 512×512 images in real-time at 500 fps. The remaining steps are implemented in software on the PC using a multi-threading approach as shown in Figure 4.6, with the following parameters. Threshold P_0 in step (1-c) was determined in order to reduce the number of feature points ($M \leq 300$), depending on the experimental scene. Step (2-a) executed 5×5 ($m = 5$) template matching with bidirectional search in the 31×31 adjacent area ($b = 31$). In step (2-b), the frame interval of feature point matching was determined with $d_0 = 7$. Step (3) executed Tukey's biweight method with $L = 10$ iterations; the parameters for the biweight evaluation functions were set at $W_l = 11 - l$ ($l = 1, \cdots, 10$).

The HFR thread is always execution at an interval of 2ms to acquire real-time processing, whereas the affine parameter accumulation and mosaicked panorama composition are conducted in composition thread to not to both the real-time processing. To monitor the panoramic videos on the display monitor, we used an independent display thread that simultaneously displays live mosaicking.

4.3 HFR Vision-based Video Stabilization

We used a Windows 7(64 bit)-OS installed PC (Hewlett Packard, Z440 workstation) with the following specifications: Intel Xeon E5-1603v4 at 2.8 GHz, 10 MB cache, 4 cores, 16 GB DDR4 RAM, two 16-lane PCI-e 3.0 buses, and four USB 3.0 ports.

4.3.1 Hardware Implementation on Spartan-3 FPGA

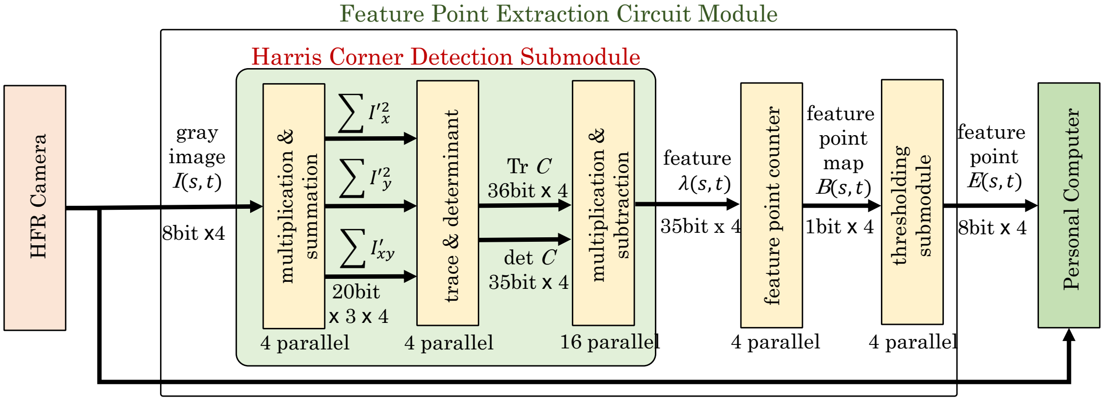


Figure 4.7: Schematic Data Flow of a Feature Point Extraction Circuit Module

On the dedicated FPGA board, as shown in Figure 4.7, the 8-bit gray-level 512×512 images could be processed in real time with circuit logic on the FPGA (Xilinx XC3S5000); the captured images and processed results could be transferred to memory allocated in the PC. The feature point detection process in step (1) was accelerated by hardware-based implementation of a feature extraction module [108] on the FPGA. The dedicated FPGA extracts feature points in a 512×512 image, and the xy coordinates of the feature points appended at the bottom 16 rows of an ROI input image of 512×496 pixels. The implemented Harris corner feature extraction module. The area size and the tunable sensitive parameter in computing the Harris corner features were set to $a = 3$ and $\kappa = 0.0625$, respectively. The area size for extracting the number of feature points was set to $p = 8$. According to the experimental scene, parameters λ_T and P_0 were determined so that the number of feature points must be less than $M = 300$.

4.3.2 Hybrid-dual Camera Video Stabilization System

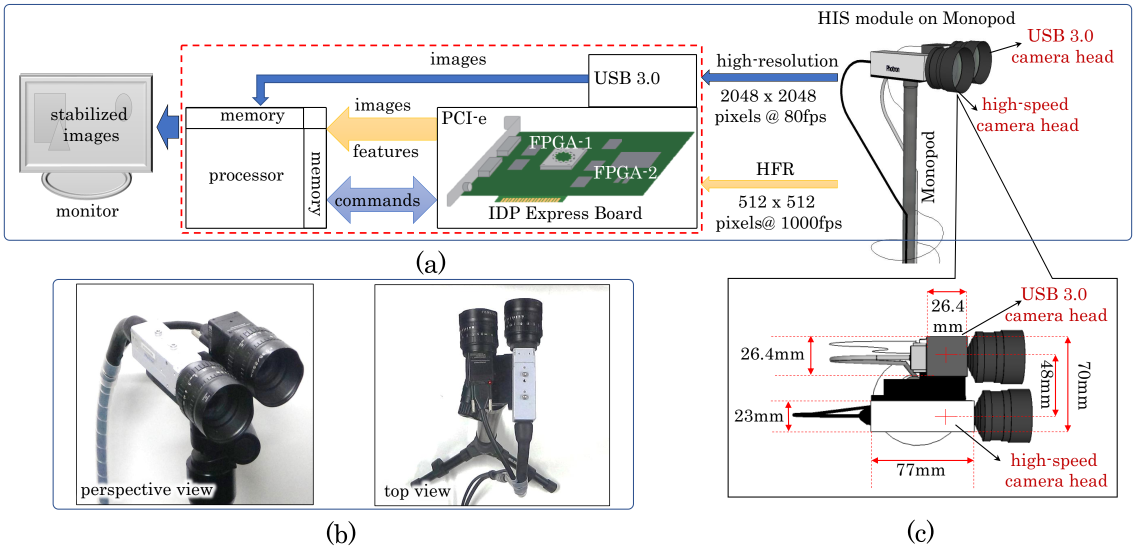


Figure 4.8: Hybrid-Dual Camera Vision System

To realize real-time high-resolution video stabilization, we implemented our algorithm on a hybrid-camera system. It consists of an FPGA-based high-speed vision device, the IDP-e [92], a high-resolution USB 3.0 camera (XIMEA, MQ042-CM), and a personal computer (PC). Figure 4.8, shows (a) the system configuration, (b) the overview of its hybrid-dual camera head when mounted on a monopod, and (c) its top-view geometric configuration. IDP-e consists of a camera head that can capture gray-level 8-bit 512×512 images at 2000 fps, and a dedicated FPGA board for hardware-implementation of the user-specific algorithms. The image sensor of the camera head is a 512×512 CMOS sensor of 5.12×5.12 mm-size at 10×10 μm-pitch. The high-resolution camera MQ042-CM can capture gray-level 8-bit 2048×2048 images, and these images can be transferred at 90 fps via a USB 3.0 interface to a PC; the sensor-size and pixel-pitch are 11.26×11.26 mm and 5.5×5.5 μm, respectively.

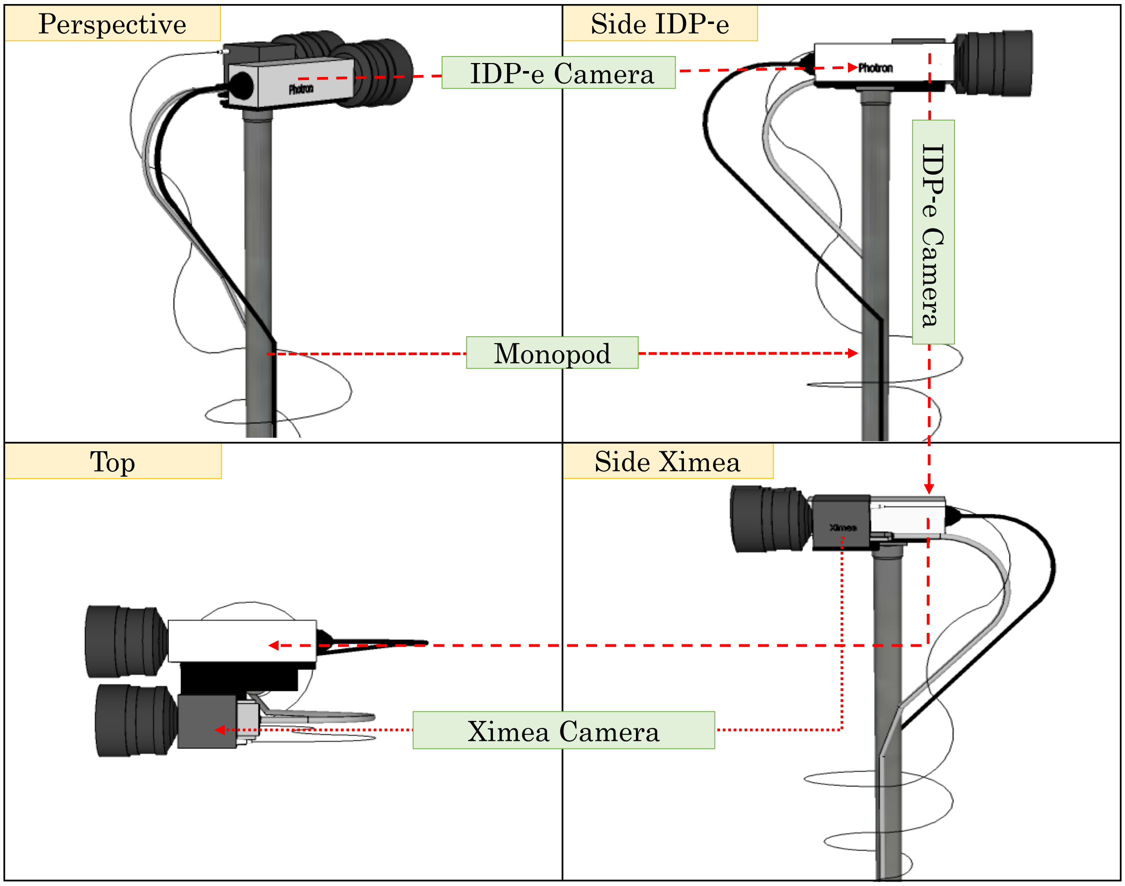


Figure 4.9: Overview of Monopod Mounted Hybrid-dual Camera Module

As shown in Figure 4.9, The hybrid-camera system was attached to a monopod of length variable from 55 to 161 cm for hand-held operation. As illustrated in Figure 4.9, the camera head of the IDP-e (camera 1) and the high-resolution camera MQ042-CM (camera 2) were installed in such a way that the optical axes of their lenses were parallel; the distance between the two axes was 48 mm. Identical CCTV lenses of $f = 25$ mm were attached to both cameras 1 and 2.

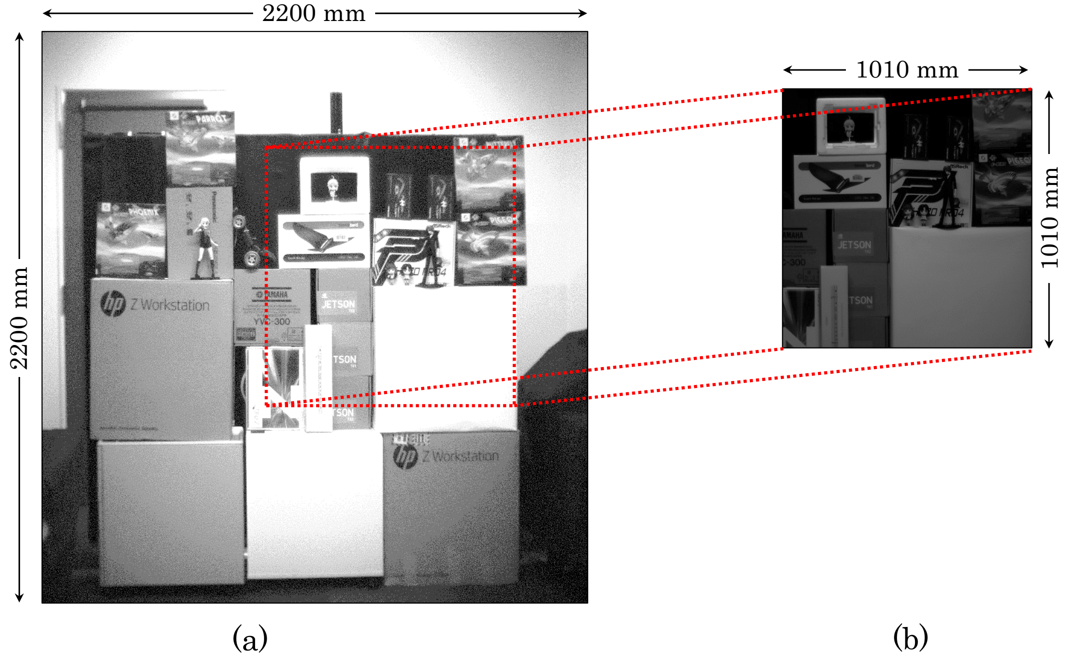


Figure 4.10: View Area of Hybrid-dual Camera Module

As shown in Figure 4.10, when the hybrid-camera module was placed 5 m away from the patterned scene (a) the high-resolution image captured by camera 2 could observe 2.20×2.20 m-area and (b) camera 1 could observe 1.01×1.01 m-area. If we observe scenes when the measurement area of camera 1 is wholly involved in that of camera 2, the high-speed vision system works as an HFR jitter sensor for stabilizing the 2048×2048 images of camera 2, as discussed earlier.

4.3.3 Software Implementation (Multi-threading) for Video Stabilization

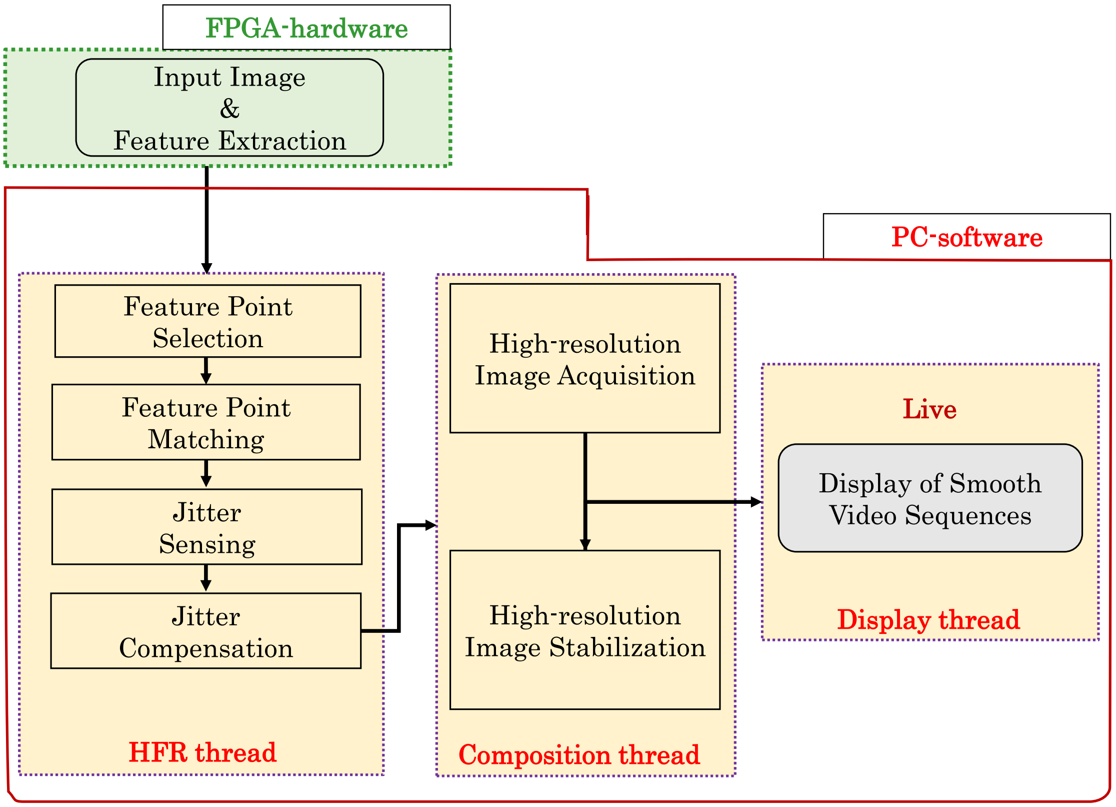


Figure 4.11: Multi-threaded Approach for Video Stabilization

Steps (2)–(4) were software-implemented on the PC. In step (2), we assumed that the number of selected feature points were less than $M = 300$, and that 5×5 ($m = 5$) template matching with bidirectional search in the 31×31 ($b = 31$) adjacent area was executed. In step (3), the high-frequency component of the frame-by-frame image displacement is extracted as its jitter displacement to be compensated by executing a 5th-order Butterworth high-pass filter ($D = 5$). The high-speed vision system captured and processed 512×496 ($N_x = 512$, $N_y = 496$) at 1000 fps, corresponding to $\Delta t = 1$ ms, whereas the high-resolution camera set for capturing 2048×2048 ($N'_x = N'_y = 2048$) at 80 fps in step (4), corresponding to $\Delta t' = 12.5$ ms. The Figure 4.11 shows the structure of the multi-threaded approach used in our stabilization application. The HFR thread is always execution at an interval of 1ms to acquire real-time processing at 1000 fps, whereas the high-resolution image acquisition at 80 fps and smooth video composition are conducted in composition thread to not to both the real-time processing of HFR thread. To monitor the high-resolution stabilized videos on the display monitor, we used an independent display thread that simultaneously displays live smooth videos.

Chapter 5

Experiments, Discussions and Confirmations

In this chapter, we demonstrate serval experiments that we have conducted to confirm the effectiveness of our implemented methods in indoor and outdoor scenarios. We also verify the consistency and performance of the approaches to compare with conventional methods. In the case of video mosaicking, we conducted motorized mosaicking experiments using a two-degree-of-freedom (2-DOF) pan-tilt actuator to compose a panoramic view of an indoor scene. We also confirm the mosaicking operation to compose a wide panoramic image of the outdoor scenic world. In the case of video stabilization, first we verified the performance of the IIR filter using a pan-actuator generating various sinusoidal frequencies. We also checked the robustness of video stabilization when mixed motion is given to the pan-tilt actuator as a combination of oscillatory motion and constant velocity from left to right direction. Finally, we demonstrate the outdoor experiment that we have conducted on stairs with a handheld hybrid-dual camera module to stabilize the high-resolution video sequence of the real-world. Further details of the experiments are stated as follows,

5.1 Mosaicked Panorama Experiments

5.1.1 Verification of Actuated Mosaicked Panorama

To confirm that the geometric consistency in panoramic images correlated with the actual camera motion, we present image mosaicking results for indoor scenes captured by a camera head mounted on a 2-DOF pan-tilt actuator system. Figure 5.1 shows the experimental setup used. The 2-DOF pan-tilt actuator system, on which the camera head of IDP-e was mounted, was fixed at the height of 1.5 m. It was able to move the camera head using the pan and tilt motors; optical encoders measured the pan and tilt angles. A C-mount 50-mm-focal-length lens was mounted on the camera head; the measurement area was 1.0 m×1.0 m at 8.3 m in front of the camera head. In the experiment, a series of wall cabinets, installed 8.3 m in front of the camera head, were observed when the pan-tilt actuator system operated (a) zigzag left-to-right camera motion with periodic up-and-down movements, and (b) accelerated left-to-right camera motion. The frame rate and exposure time of the IDP-e were set to 500 fps and 2 ms, respectively. For the selection of 100 feature points or less from a 512×512 image, the threshold parameters were set to $\lambda_T = 5 \times 10^7$ and $P_0 = 25$.

5.1.1.1 Experimental Setup

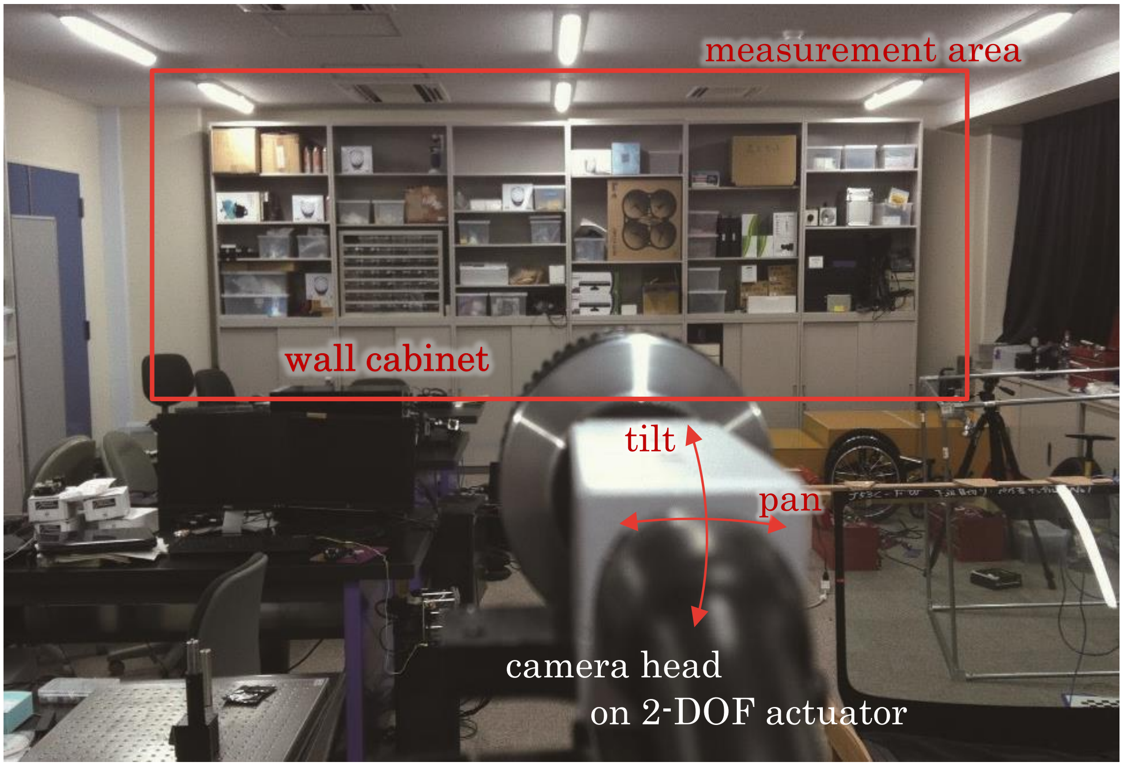


Figure 5.1: Experimental Setup for Motorized Mosaciking

5.1.1.2 Zigzag Displacement path

Figure 5.2 shows the pan and tilt angles of pan-tilt actuator, the x- and y-coordinates for translation vector, the number of extracted feature points, and the frame interval of feature point matching for $t = 0.00$–1.30 s obtained when the camera head was moved from left to right with periodic up-and-down motions three times every two seconds.

The affine matrix and translation vector were set at unit matrix and $(255, 499)$, respectively, at $t = 0$. The x coordinate gradually increased from 255 to 2776 pixel for $t = 0.00$–1.30 s as the pan angle increased from $-9.6°$ to $23.2°$; the y coordinate changed periodically at 1.5 Hz in the range between 500 and 1000 pixel, corresponding to the periodically changing tilt angle.

The interval of feature point matching was adjusted to within 4 ms because there were always large image displacements between frames in the zigzag camera motion. Figure 5.3 shows the synthesized panoramic images of 3200×1200 pixels, taken at intervals of 0.12 s. In the 3200×1200 panoramic images, $(x, y) = (0, 0)$ corresponded to their left-upper corners. Over time, the panoramic image was extended by accurately stitching affine transformed input images over the panoramic image at the previous frame, and a series of wall cabinets were observed in a single panoramic image at $t = 1.26$ s.

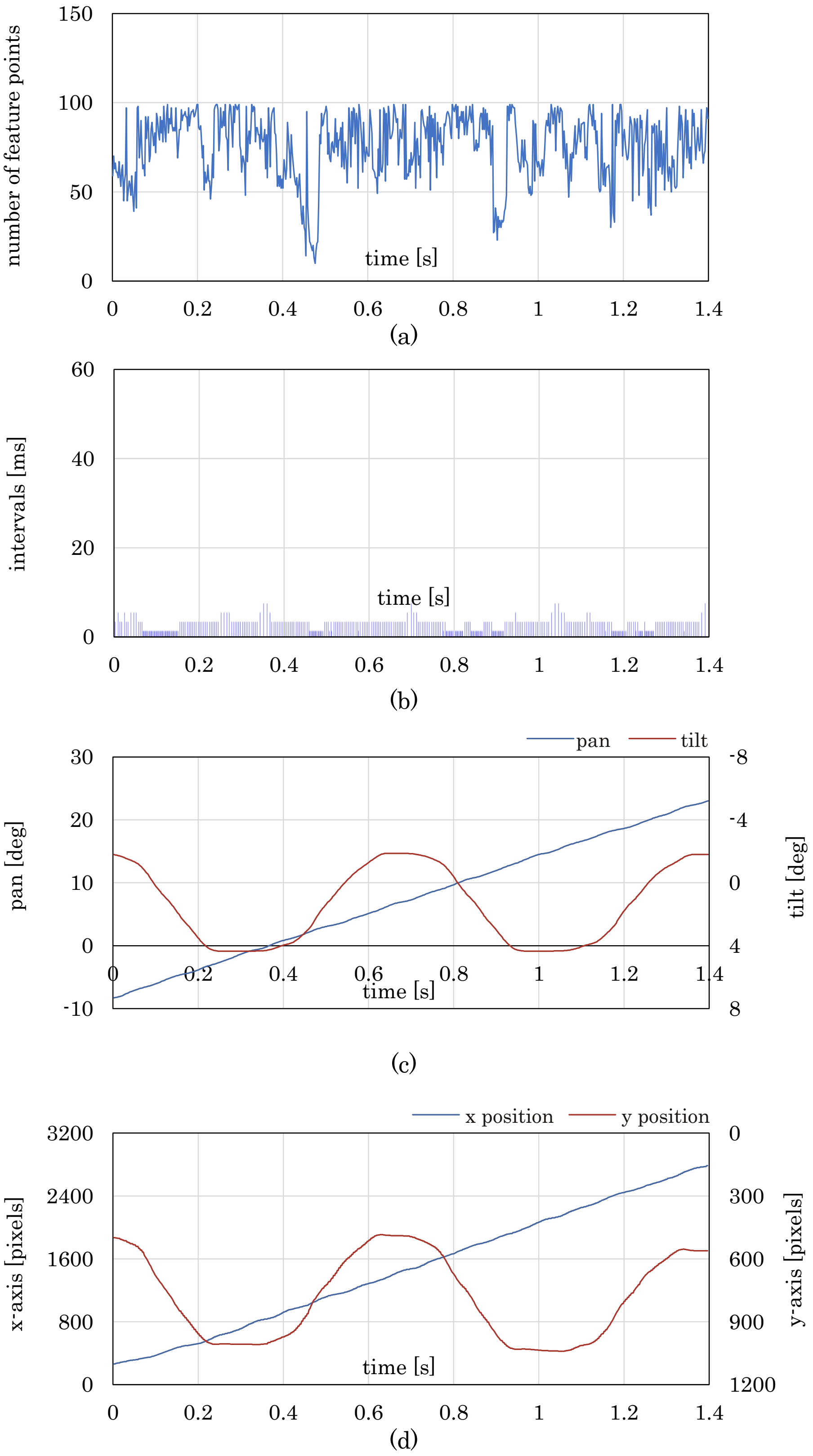
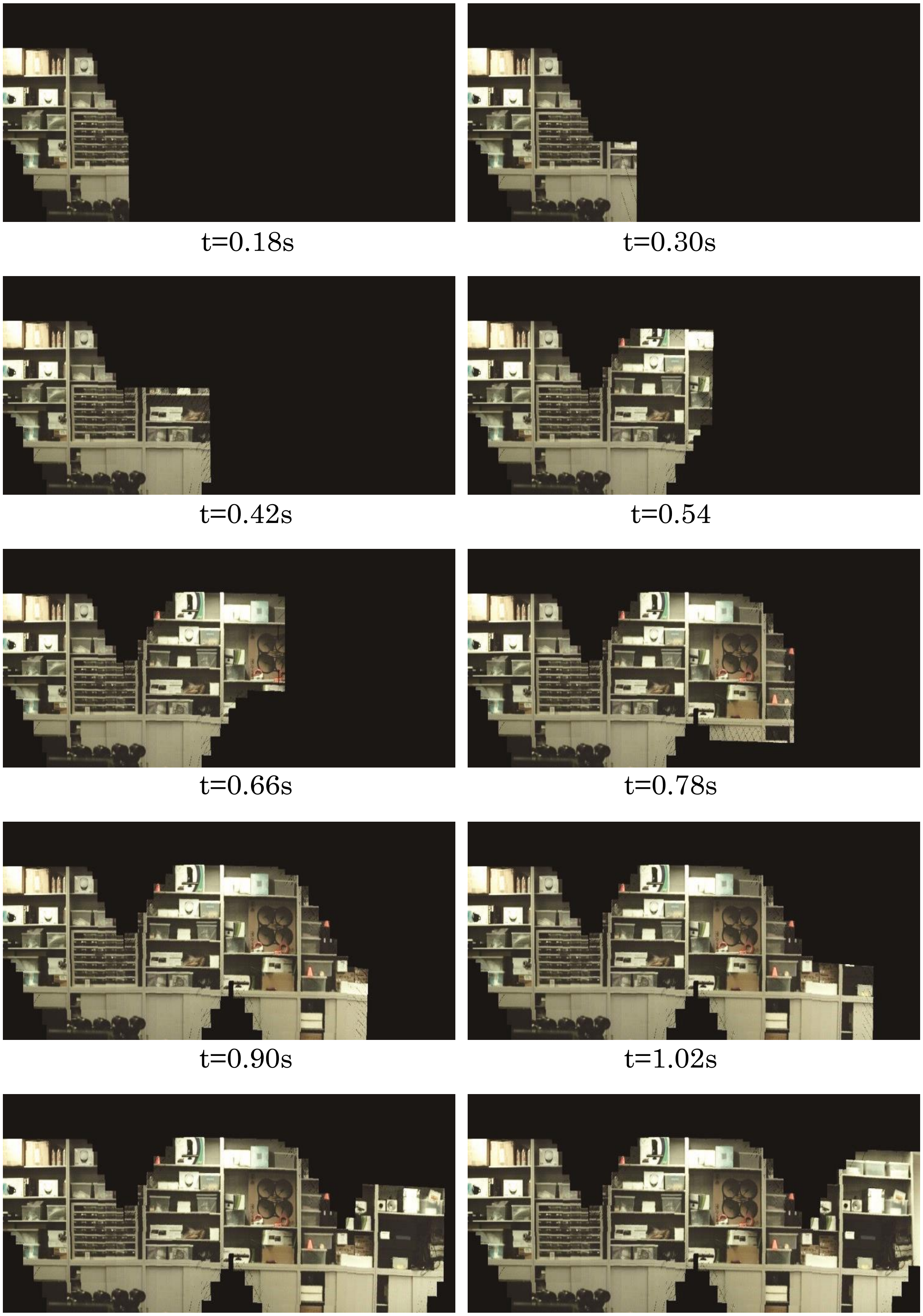


Figure 5.2: Graph Showing (a)Feature Point Extraction, (b) Frame Interval, (c) Pan-Tilt Angles of 2-DOF Actuator, and (d) x-y Displacement Trajectory of Camera



t=0.18s

t=0.30s

t=0.42s

t=0.54

t=0.66s

t=0.78s

t=0.90s

t=1.02s

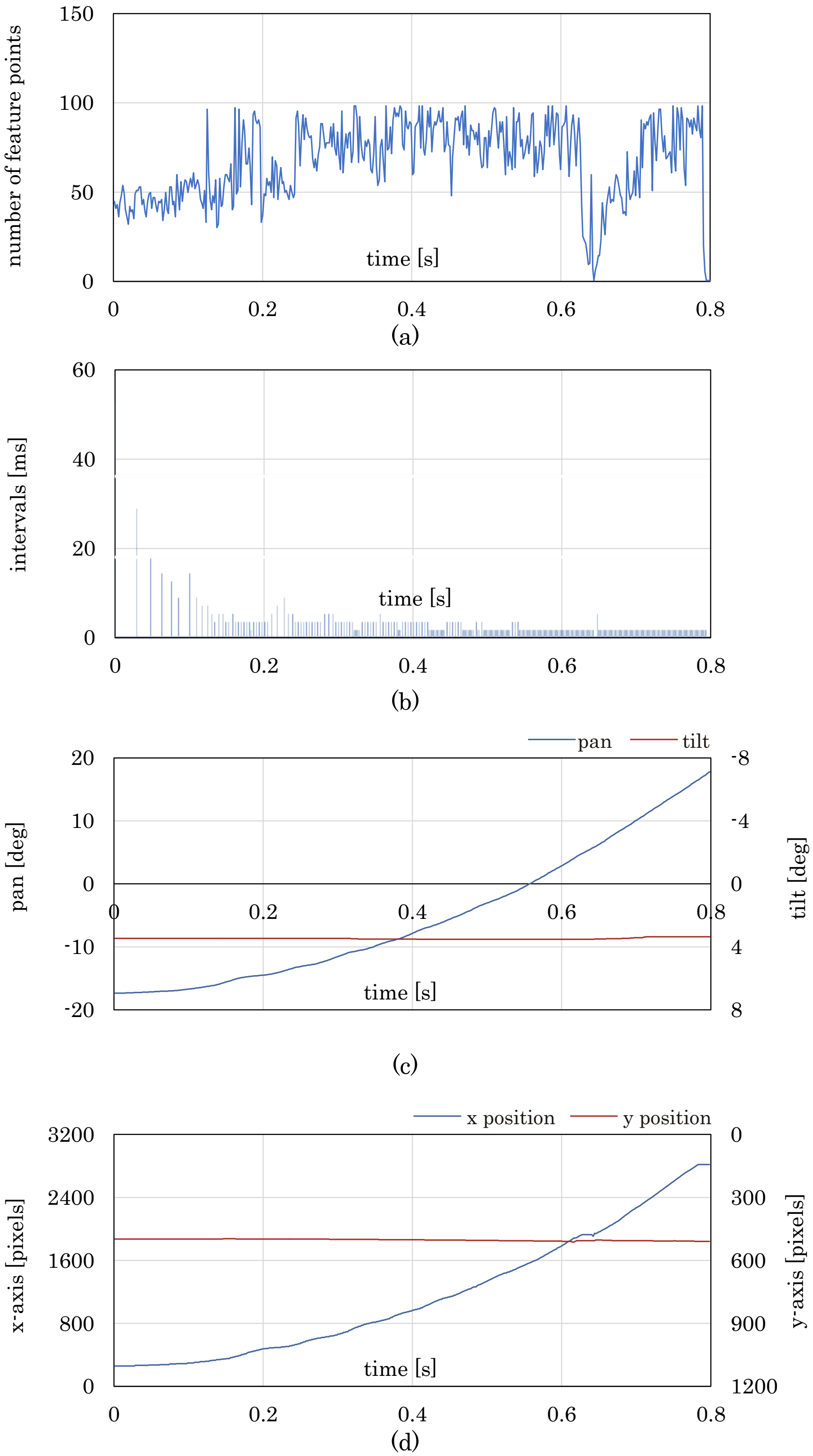
t=1.14s

t=1.26s

Figure 5.3: Mosaicked Panorama Composed with Motorized Zigzag Camera Motion

5.1.1.3 Accelerated Displacement path

Figure 5.4 shows the pan and tilt angles of pan-tilt actuator, the x- and y-coordinates for translation vector, the number of extracted feature points, and the frame interval of feature point matching for $t = 0.00$–0.86 s obtained for unidirectional camera head movement from left to right with increasing speed. The value of the x coordinate increased as the pan angle increased, whereas the value of the y coordinate remained constant due to no change in the tilt angle. With respect to the changes in the pan angles, a long frame interval of dozens of milliseconds was set for the slow camera motion around $t = 0$, and a short frame interval of 2 ms for the fast one as time passed. Figure 5.5 shows the synthesized panoramic images of 3200×1200 pixels, taken at intervals of 0.09 s. Over time, the panoramic image was also extended by accurately stitching together transformed input images. The number of feature points was always less than 300 points; the frame interval was adjusted to correspond to the 2-Hz periodic change in the scene observed in the input images. Based on the motion of the camera, it can be seen that a short frame interval was set for the high-speed scene.



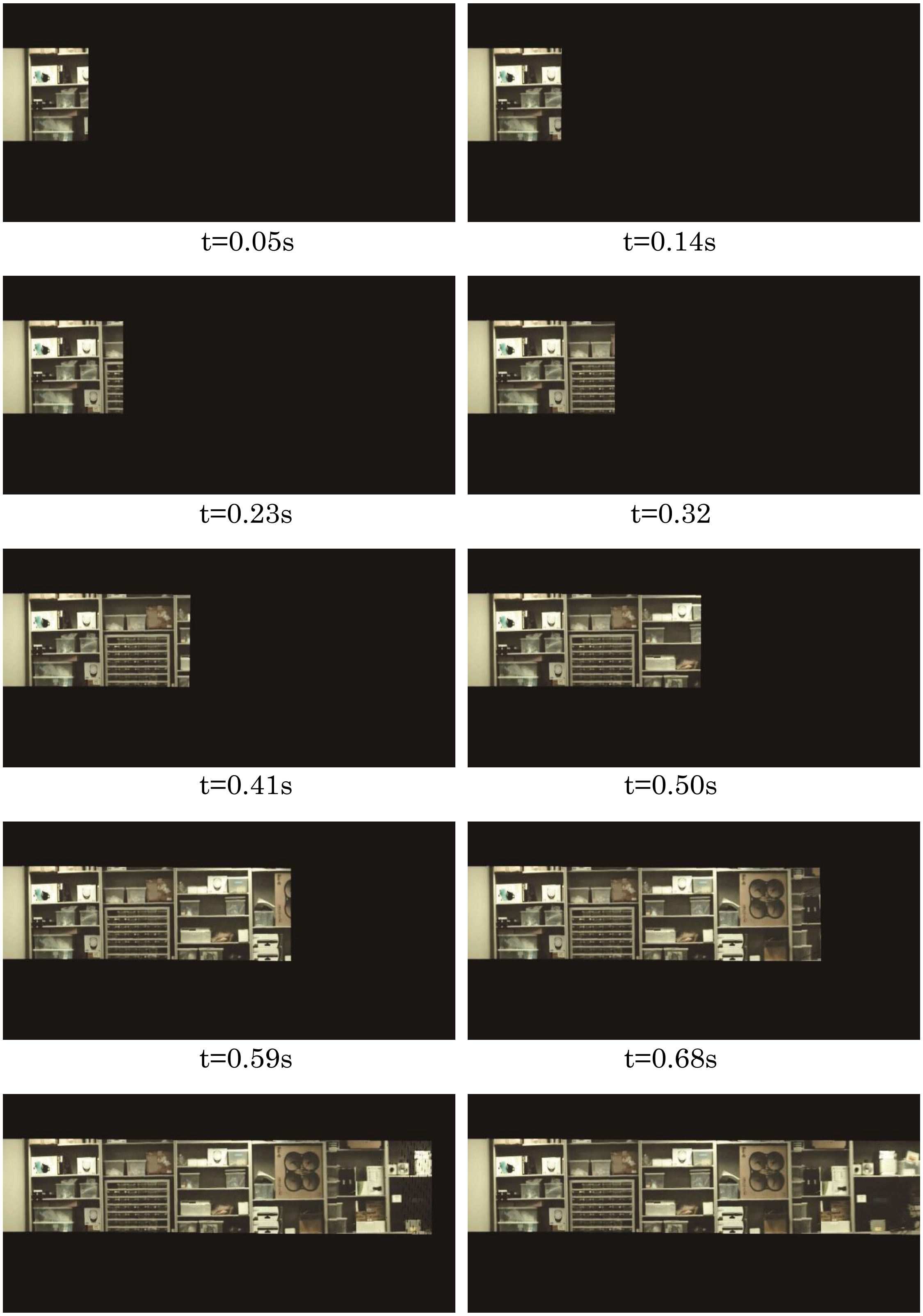
(a)

(b)

(c)

(d)

Figure 5.4: Graph Showing (a)Feature Point Extraction, (b) Frame Interval, (c) Pan-Tilt Angles of 2-DOF Actuator, and (d) x-y Displacement Trajectory of Camera



t=0.05s

t=0.14s

t=0.23s

t=0.32

t=0.41s

t=0.50s

t=0.59s

t=0.68s

t=0.77s

t=0.86s

Figure 5.5: Mosaicked Panorama Composed with Motorized Accelerated Camera Motion

5.1.2 Real-world Outdoor Experiment of Video Mosaicking

Next, we present the image mosaicking result for an outdoor scene captured by a camera head moved quickly and periodically by a human hand. Figure 5.6 shows the experimental outdoor scene.

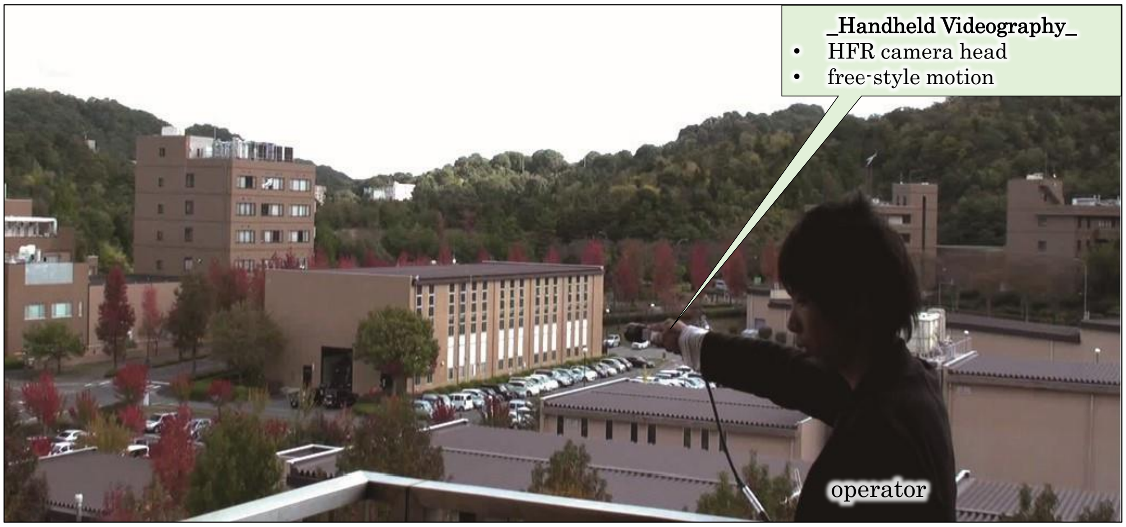


Figure 5.6: Handheld Experimental Setup

The scene was captured from the top of an eight-story building in Hiroshima University by an operator who manually moved the hand-held camera head with his hand from left to right with periodic up-and-down motions approximately five times every three seconds, and then decelerated the movement of the HFR camera. In the experiment, a C-mount 25-mm-focal-length lens was mounted on the camera head. The frame rate and exposure time of the IDP Express system were set to 500 fps and 0.25 ms, respectively. To select the 300 feature points or less from a single 512×512 image, the threshold parameters were set to $\lambda_T = 5 \times 10^7$ and $P_0 = 25$.

Figure 5.7 shows the five-input image sequence and the extracted feature points, taken at intervals of 0.2 s. It can be seen that most of the feature points in the input images were correctly extracted.

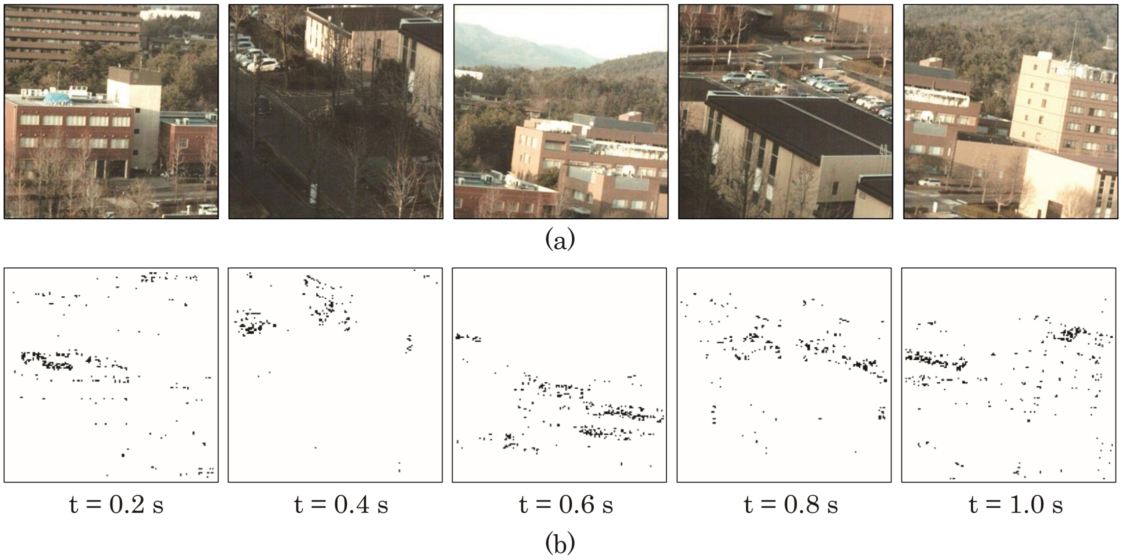
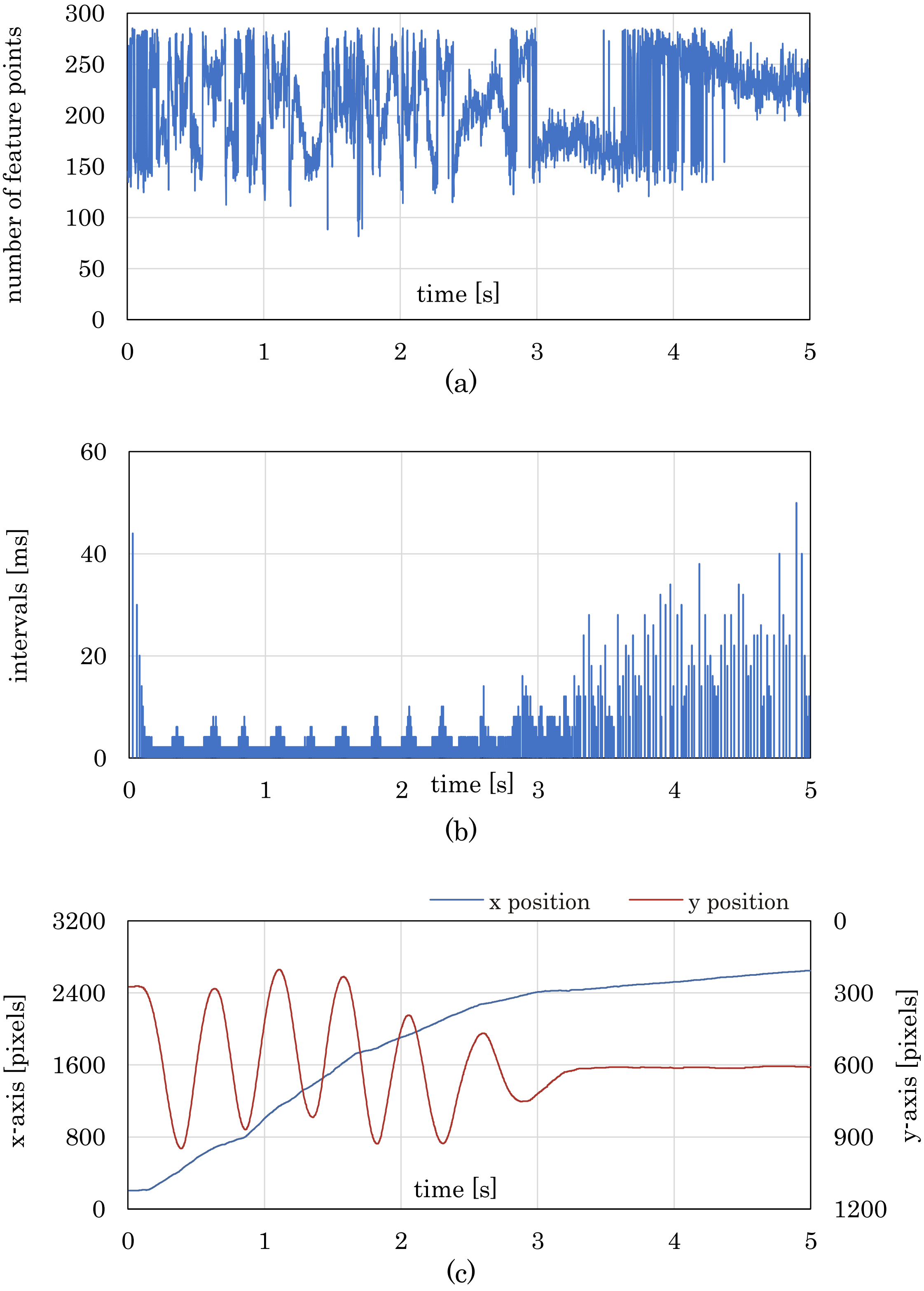


Figure 5.7: (a) Input Images and (b) Feature Point Extracted from HFR Sequence

Figure 5.8 shows the number of extracted feature points, the frame interval of feature point matching, and the x- and y-coordinates for the translation vector for $t = 0.0$–5.0 s. Based on the motion of an HFR camera, a short frame interval was set for the high-speed scene and a long frame interval for the low-speed scene.

Figure 5.9 shows the synthesized 3200×1200 panoramic images, taken at intervals of 0.5 s. Over time, the panoramic image was extended by stitching together transformed images over the panoramic image at the previous frame, and large buildings and background forests on the far side were observed in a single panoramic image at $t = 4.5$ s. These figures indicate that our system can accurately generate a single synthesized image in real-time by stitching together 512×512 input images at 500 fps when the camera head is quickly moved by a human hand.

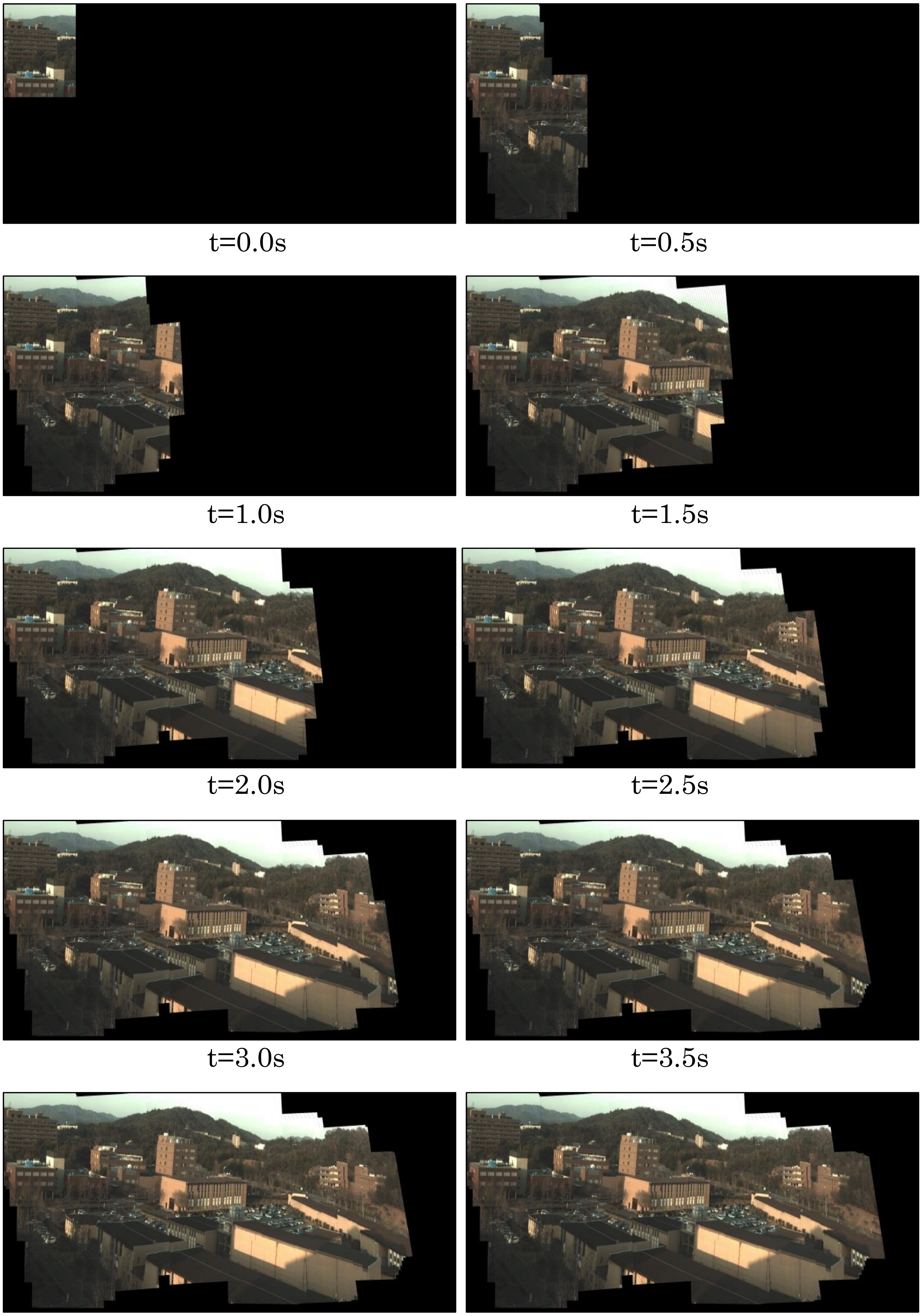


(a)

(b)

(c)

Figure 5.8: Graph Showing (a)Feature Point Extraction, (b) Frame Interval, and (c) x-y Displacement Trajectory of Camera



t=0.0s

t=0.5s

t=1.0s

t=1.5s

t=2.0s

t=2.5s

t=3.0s

t=3.5s

t=4.0s

t=4.5s

Figure 5.9: Real-world Scene Handheld Video Mosaicking

5.1.3 Discussions

In this section, we report the development of a real-time image mosaicking system that can generate panoramic images from 512×512 images captured at 500 fps by implementing an accelerated feature-based image mosaicking algorithm on an FPGA-based high-speed vision platform.

5.1.3.1 Computational Efficiency

	time
(1-a,1-b) Feature extraction / Thresholding	0.86
(1-c) Selection of feature points	0.05
(2-a) Template matching	0.97
(2-b) Frame interval selection	0.01
(3) Affine transform estimation	0.09
(4) Image composition	51.43
Total ((1)–(3))	1.98

Table 5.1: Execution times on HFR Vision-based Video Mosaicking System (unit: ms).

Table 5.1 summarizes the execution times of steps (1)–(4) for image mosaicking of 512×512 images when our image mosaicking algorithm was implemented on the high-speed vision platform with the parameters stated above. The execution times of steps (1-a) and (1-b) include the image acquisition time for a 512×512 image on the FPGA board. The total execution time of steps (1), (2), and (3) was less than 2 ms; it was worse when step (3) was executed at all the frames. In this study, the shortest frame interval in the system was set to Δt = 2 ms. The execution time of step (4) was much larger than that of the other steps. Step (4) was implemented as a multi-threaded process that was not able to disturb the real-time processes of the other steps when its interval was set to $\Delta t'$ = 60 ms for monitoring by the human eye. In this way, we confirmed that image mosaicking, including feature point extraction and tracking for $M \leq 300$, can be executed for 512×512 images in real-time at 500 fps. The upper bound of feature points number is 300 to guarantee our system working at 500 fps, which vary from PC configuration.

5.2 Video Stabilization Experiments

5.2.1 Verification of Jitter Filtering

5.2.1.1 Experimental Setup and Parameters

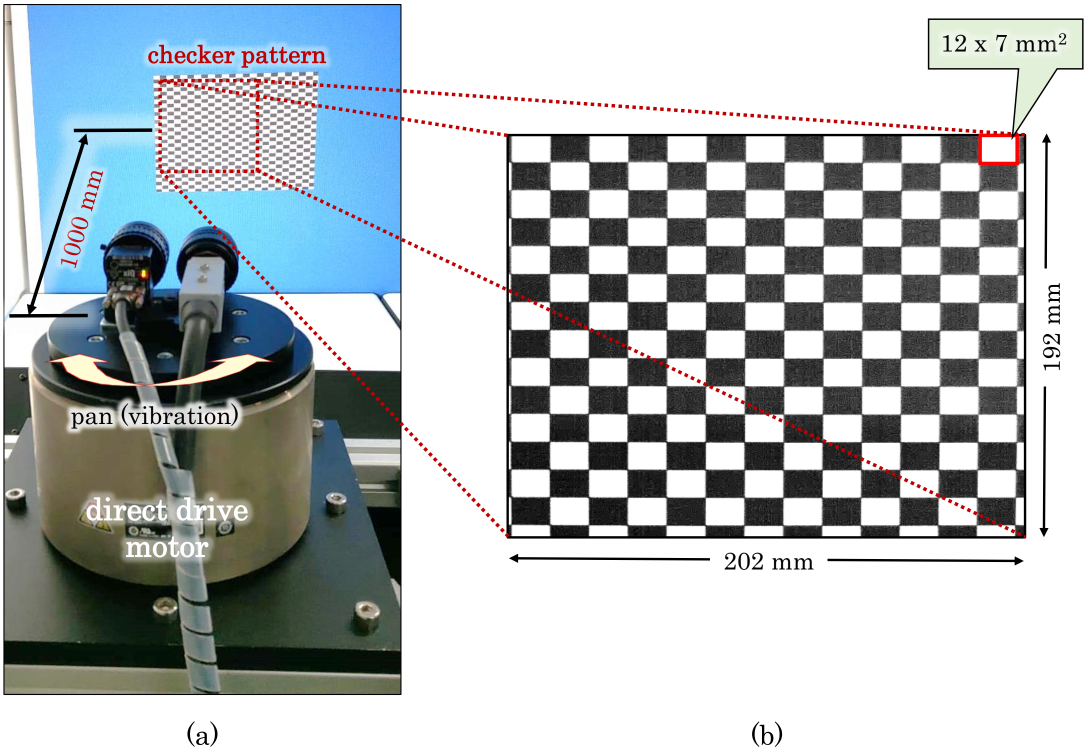


Figure 5.10: Experimental Setup for Filter Response Verification

Firstly, we evaluated the performance of our system in video stabilization by observing a static checkered pattern when a hybrid-camera system vibrated mechanically in the pan direction, as illustrated in Figure 5.10. The hybrid-camera system was mounted on a direct drive AC servo motor (Yaskawa, SGM7F-25C7A11) so as to mechanically change its pan angle, and a 12×7 mm-pitch checkered pattern was installed 1000 mm in front of the camera system. The measurement area observed in the 512×496 image of camera 1 corresponded to 202×192 mm on the checkered pattern.

5.2.1.2 IIR-5th Order High Pass Filter Response

In the pan direction, the hybrid-camera system vibrated on the 2.5 degree-amplitude sinusoid trajectories at various frequencies ranging from 0.1 Hz to 3 Hz with increments of 0.1 Hz.This camera ego-motion exhibited 120-pixel displacement in the horizontal direction in the camera 1 image. The threshold parameters in the feature extraction step were set to $\lambda_T = 5 \times 10^7$ and $P_0 = 15$, and the cut-off frequency in the jitter sensing step was set to $f_{cut} = 0.5$ Hz. Figure 5.11 shows (a) the response of a 5th-order Butterworth high-passfilter, of which the cut-off frequency is 0.5 Hz, and (b) the relationship between the vibration frequencies and the damping ratios in the jitter cancellation on our system .

(a) Filter Coefficients:

We derived the coefficients for the 5th-Order Butterworth High Pass filter for 1 ms sampling time with 0.5 Hz cut-off frequency by using the Matlab digital filter toolbox. The numerator and denominator coefficients for the filter are tabulated in the Table 5.2. We used these coefficients in our software program to process sensed displacements to eliminate the motion anomalies in the sequences.

$$\begin{aligned}\boldsymbol{y_0} = {} & ((\boldsymbol{b}_0)(\boldsymbol{x}_0) - (\boldsymbol{a}_5)(\boldsymbol{y}_5) - (\boldsymbol{a}_4)(\boldsymbol{y}_4) - (\boldsymbol{a}_3)(\boldsymbol{y}_3)) \\ & - ((\boldsymbol{a}_2)(\boldsymbol{y}_2) - (\boldsymbol{a}_1)(\boldsymbol{y}_1) + (\boldsymbol{b}_5)(\boldsymbol{x}_5) + (\boldsymbol{b}_4)(\boldsymbol{x}_4)) \\ & + ((\boldsymbol{b}_3)(\boldsymbol{x}_3) + (\boldsymbol{b}_2)(\boldsymbol{x}_2) + (\boldsymbol{b}_1)(\boldsymbol{x}_1)),\end{aligned} \tag{5.1}$$

Where $\boldsymbol{y_0}$ high pass filter output, $\boldsymbol{y_1}$-$\boldsymbol{y_5}$ are the filter feedback and $\boldsymbol{x_1}$-$\boldsymbol{x_5}$ are the input samples.

We used the following coefficients b and a in the 5.1 equation to filter the input motion anomalies available in the camera displacements. $\boldsymbol{y_0}$ is outputted as a remaining high frequency component.

coefficient	value
(1) b0	0.99492969135382414
(2) b1	-4.9746484567691205
(3) b2	9.9492969135382427
(4) b3	-9.9492969135382427
(5) b4	4.9746484567691205
(6) b5	-0.99492969135382414

(a) b-coefficients (numerator)

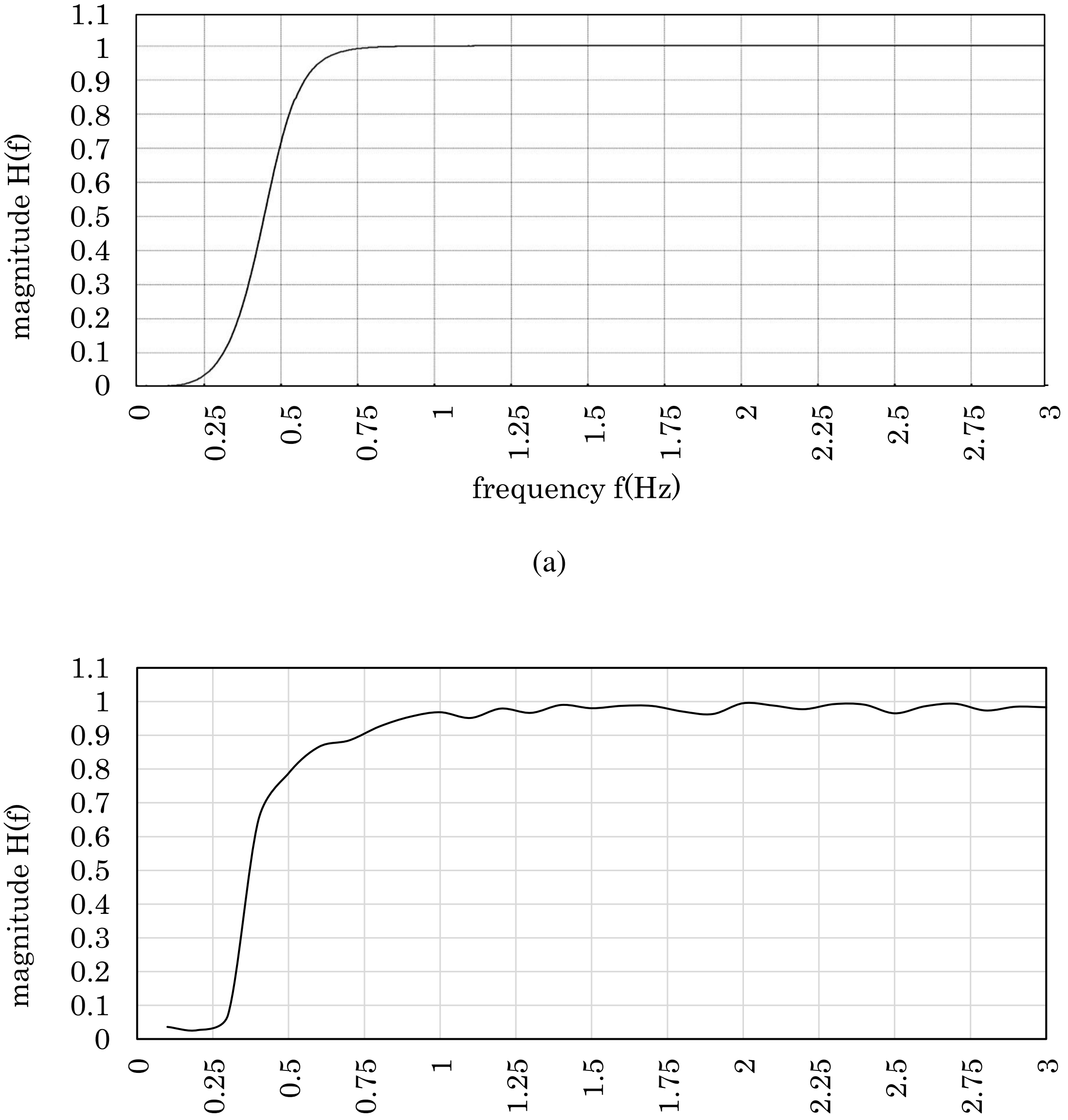
coefficient	value
(1) a0	1
(2) a1	-4.9898335938352982
(3) a2	9.9593860340854388
(4) a3	-9.9391563770883185
(5) a4	4.9594890275758976
(6) a5	-0.98988509073741515

(b) a-coefficients (denominator)

Table 5.2: Coefficients of 5th-Order IIR Butterworth High Pass Filter

(b) Filter Response:

The pan-actuator oscillated with various sinusoidal frequencies ranging from 0.1 Hz to 3.0 Hz to verify the response of the implemented filter with the ideal response obtained from the Matlab digital filter toolbox. The damping ratio was computed as a ratio of the standard deviation of the filtered high-frequency component to that of the jitter displacement in the horizontal direction for 10 s for all the frequencies mentioned above. As shown in Figure 5.11-(a) the ideal response of the 5th-Order IIR Butterworth High Pass Filter, and (b) the response of the practical filter that we have implemented. It is observed that both the responses are similar; hence our digital filter is verified and suitable for the stabilization application.

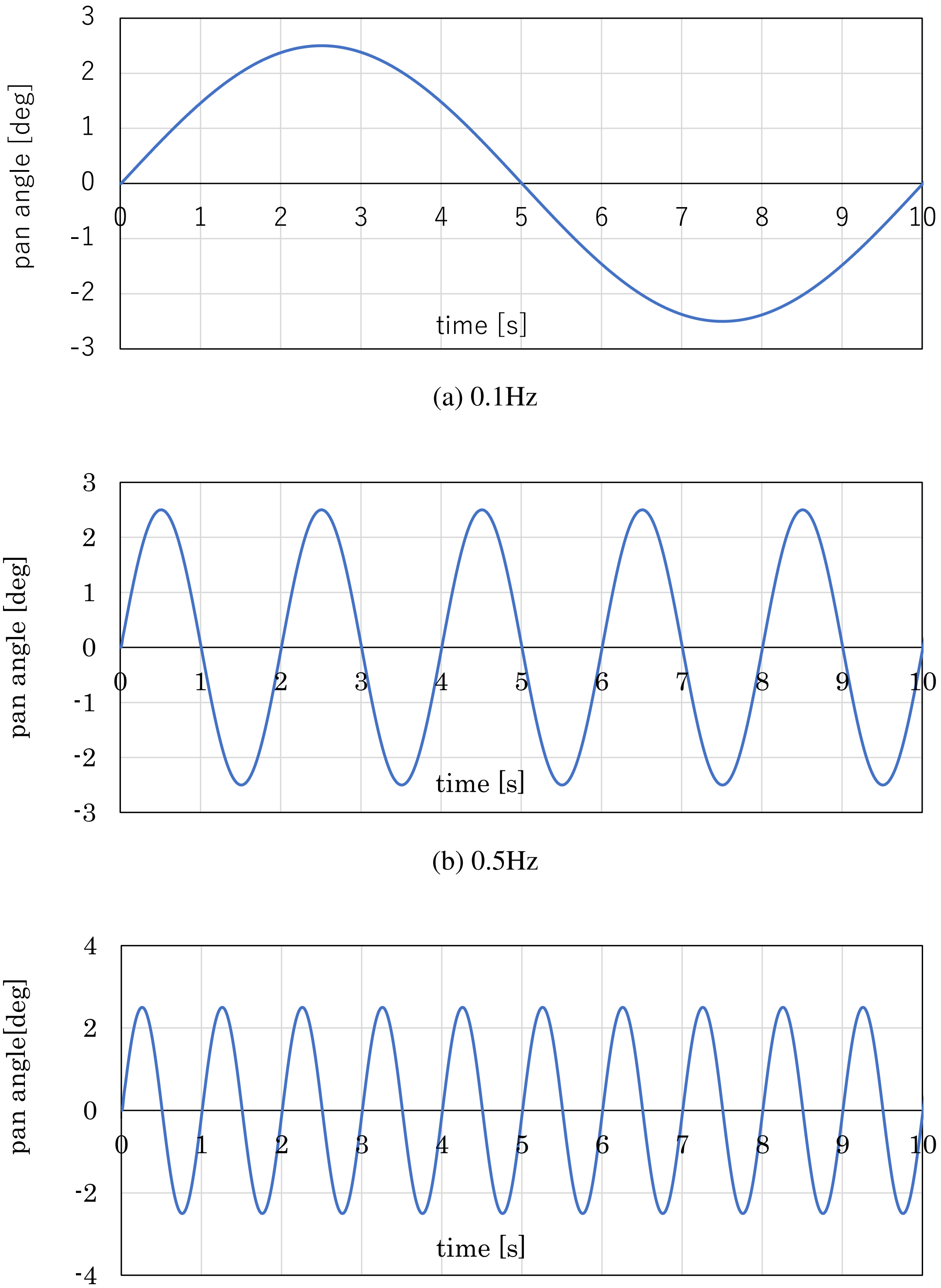


(a)

(b)

Figure 5.11: Graph showing (a) Ideal Filter Response and (b) Practical Response of 5th-Order IIR Butterworth High pass Filter

(c) Filter performance for Various Frequencies:

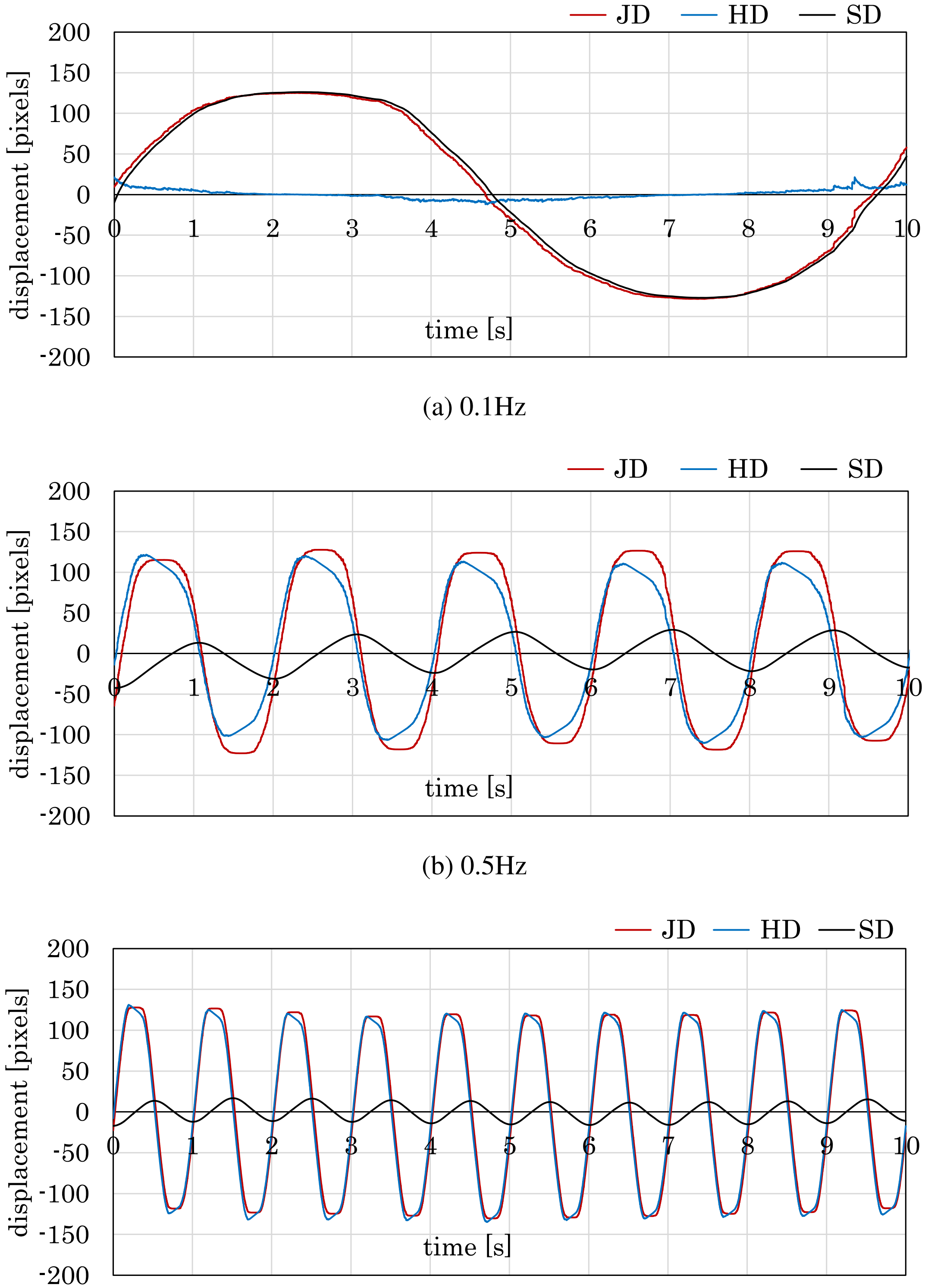


(a) 0.1Hz

(b) 0.5Hz

(c) 1.0Hz

Figure 5.12: Graph Showing Frequencies Generated by Pan-actuator



(a) 0.1Hz

(b) 0.5Hz

(c) 1.0Hz

Figure 5.13: Graph Showing Response of 5th-Order IIR Butterworth High pass Filter to Various Frequencies

Figure 5.12 graphs show the motion generated by the pan-actuator for three selected frequencies as, 0.1, 0.5 and 1.0 Hz, logged by the optical encoder of the actuator. Figure 5.13 shows the the jitter displacements (JDs), their filtered high-frequency component displacements (HDs), and the stabilized displacements (SDs) in the horizontal direction in the camera 1 image for 10 s when the hybrid-camera system was vibrated at 0.1, 0.5, and 1.0 Hz. The SDs were computed by canceling the HDs from the JDs. According to the cut-off frequency at 0.5 Hz, it can be observed that the SD was almost matched with the JD when the vibration was done at 0.1 Hz, whereas the SD tended toward zero with 1.0 Hz vibration. This tendency can be confirmed in Figure 5.11-(b) where it can be observed that our system detected and canceled a specified high-frequency camera jitter displacement and that the damping ratio was largely varied from 1 to 0 around the cut-off frequency f_{cut} = 0.5 Hz.

5.2.2 Mixed Motion Video Stabilization

We evaluated the video stabilization performance by observing a printed photographic pattern when the hybrid-camera system moved with drifting at a certain frequency in the pan direction. The mixed movement was given to the actuator, which consisted of constant velocity from left to right and sinusoidal oscillation.

5.2.2.1 Experimental Setup and Parameters

A printed cityscape photographic pattern of dimensions 1200×900 mm was placed 1000 mm in front of the hybrid-camera system mounted on a pan-tilt motor head as shown in Figure 5.14. Figure 5.15 shows the (a) area observed on the pattern, the 440×440 mm by camera 2 involved and (b) the 202×192 mm by camera 1.

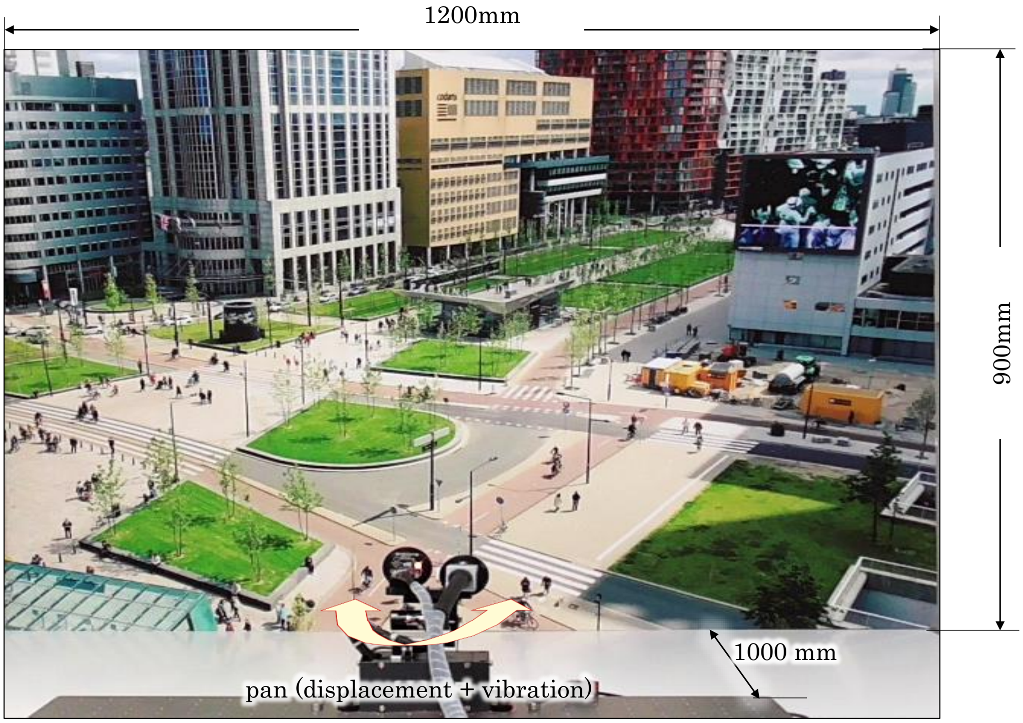


Figure 5.14: Experimental Setup for Motorized Mixed Motion Generator

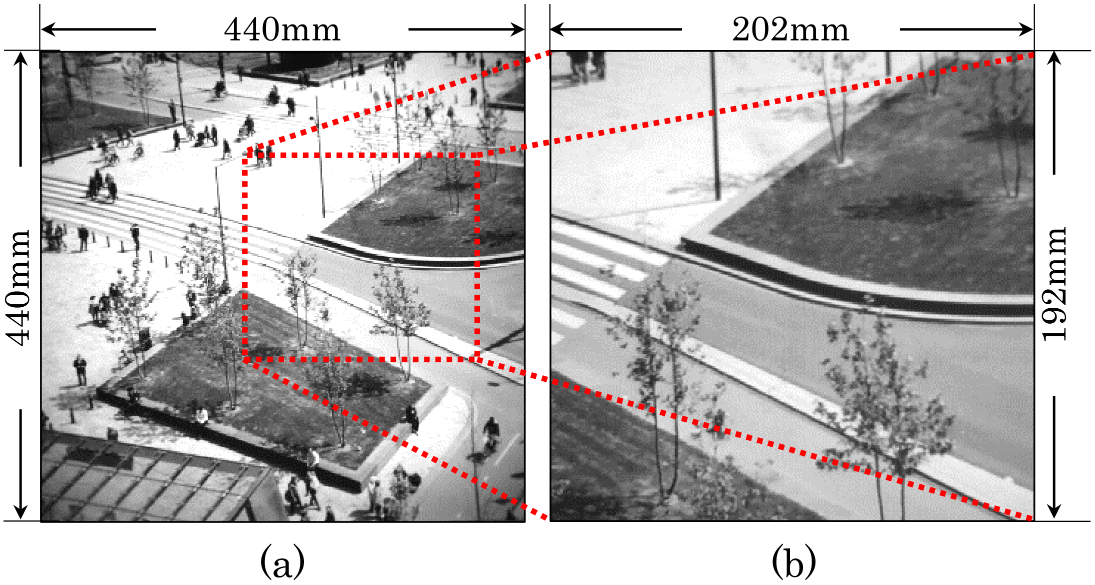
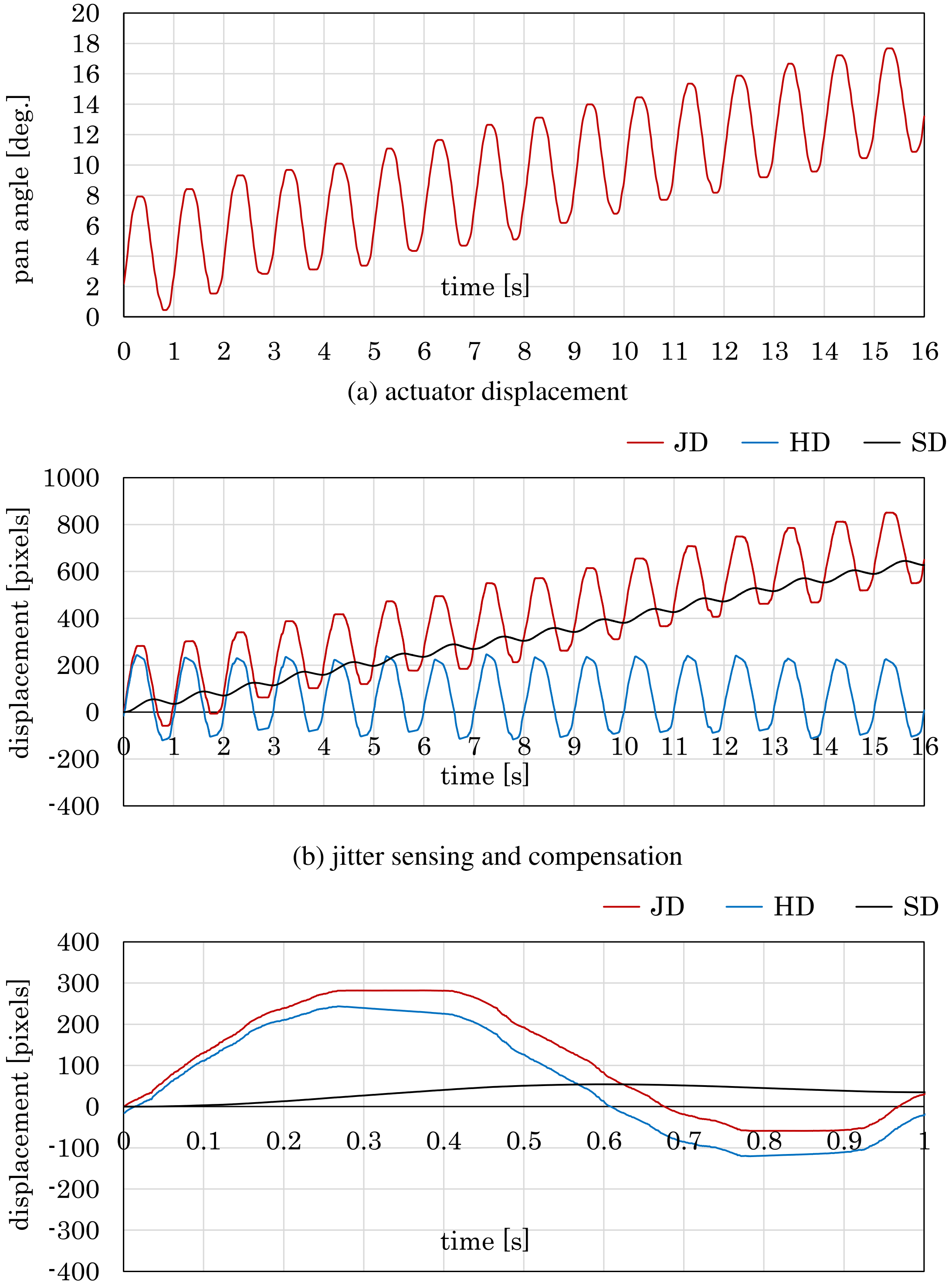


Figure 5.15: View Area by Camera 2 with Overlap of Camera 1

5.2.2.2 Confirmation of Stabilization of High-resolution Image Sequences

In the experiment, the pan angle varied with 1 Hz vibration as illustrated in Figure 5.16-(a). The parameters in the feature extraction step and the cut-off frequency in the jitter sensing step were set to the same values as those in the previous subsection. Figure 5.16-(b) shows the JD, the HD, and the SD in the horizontal direction in the camera 1 image for 16 s when the hybrid-camera system drifted with 1 Hz vibration in the pan direction shown in Figure 5.16-(c). In Figure 5.17, (a) the extracted feature points (green '+'), (b) selected feature points (red '+'), and (c) the pairs of the matched feature points between previous and current frames (blue and red dots) were plotted in the 512×496 input images of camera 1. Figure 5.18 shows (a) the 2048×2048 input images and (b) the stabilized images of camera 2.

The images in Figures 5.17 and 5.18 were taken for t = 0–14 s with an interval of 2 s. Figure 5.18 shows (a) the 2048×2048 input images and (b) their stabilized images of camera 2 from t = 0 to 0.7 s, taken at intervals of 0.1 s. In Figure 5.16, the SD was obtained as the DC component by reducing the 1 Hz vibration in the camera drift, which is higher than the cut-off frequency of 0.5 Hz. The stabilized images of camera 2 for 0.7 s in Figure 5.18(b) were compensated to cancel the 1 Hz vibration, whereas the apparent left-to-right motion of the cityscape scene for 14 s in the stabilized images of camera 2, which corresponded to the DC component in the camera drift, was not canceled, as illustrated in Figure 5.18(b). These experimental results show that our hybrid-camera system can automatically stabilize 2048×2048 images of complex scenes to cancel high-frequency components in the camera ego-motion.

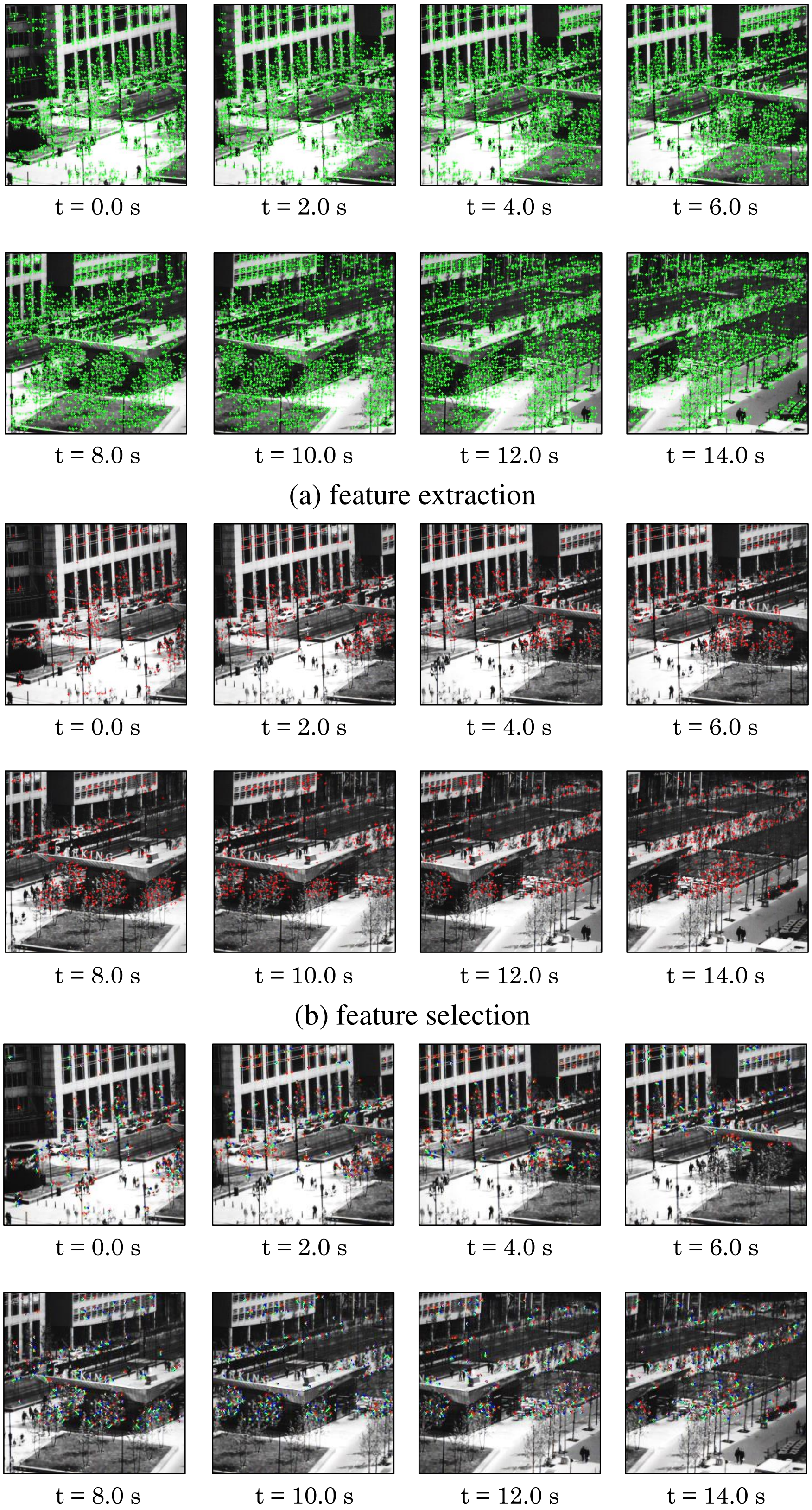


(a) actuator displacement

(b) jitter sensing and compensation

(c) 1.0Hz jitter and its compensation

Figure 5.16: Graph Showing Stabilization of Jitter Displacements Conducted using Motorized Mixed Motion

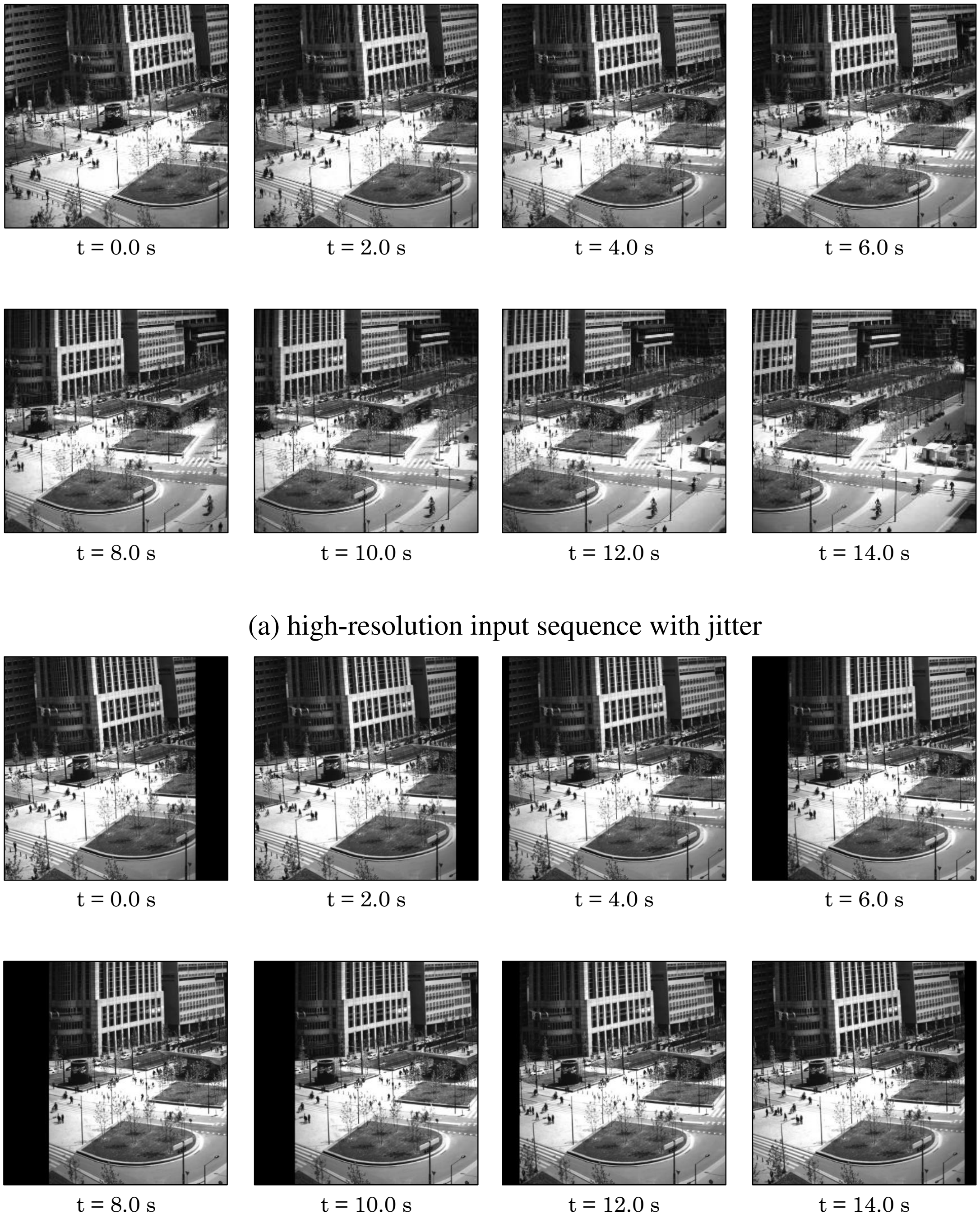


(a) feature extraction

(b) feature selection

(c) feature matching

Figure 5.17: HFR Vision-based Jitter Sensing Conducted using Motorized Mixed Motion



(a) high-resolution input sequence with jitter

(b) high-resolution stabilized sequence with reduced jitter

Figure 5.18: Video Stabilization of High-resolution Jittered Sequence Conducted using Motorized Mixed Motion

5.2.3 Real-world Outdoor Experiment of Video Stabilization

To demonstrate the performance of our proposed system in a real-world scenario, we conducted an experiment when an operator was holding a hand-held dual-camera head of our hybrid-camera system while walking on outdoor stairs where undesired camera ego-motion usually induces unpleasant jitter displacements in video shooting.

5.2.3.1 Experimental Setup and Parameters

Figure 5.19 shows the experimental scene when walking down on outdoor stairs holding a dual-camera head. It was mounted on a 70 cm-long monopod. In the experiment, we captured an outdoor scene ofwalking multiple persons with background trees; they were walking on the stairs at a distance of 2 to 4 m from the operator. Induced by left-and-right hand-arm movement and up-and-down body movement while walking, the dual camera head was repeatedly panned in the horizontal direction and moved in the vertical direction around 1 Hz.At a distance of 3 m from the operator, an area of 1.30×1.30 m corresponded to a 2048×2048 input image of camera 2, which involved an area of 0.60×0.55 m observed in a 512×496 image of camera 1. The threshold parameters in the feature extraction step were set to $\lambda_T = 5 \times 10^7$ and $P_0 = 15$, and $M = 300$ feature points or less were selected for feature point matching. The cut-off frequency in the jitter sensing step was set to $f_{cut} = 0.5$ Hz. to reduce the 1 Hz camera jitter in the experiment.

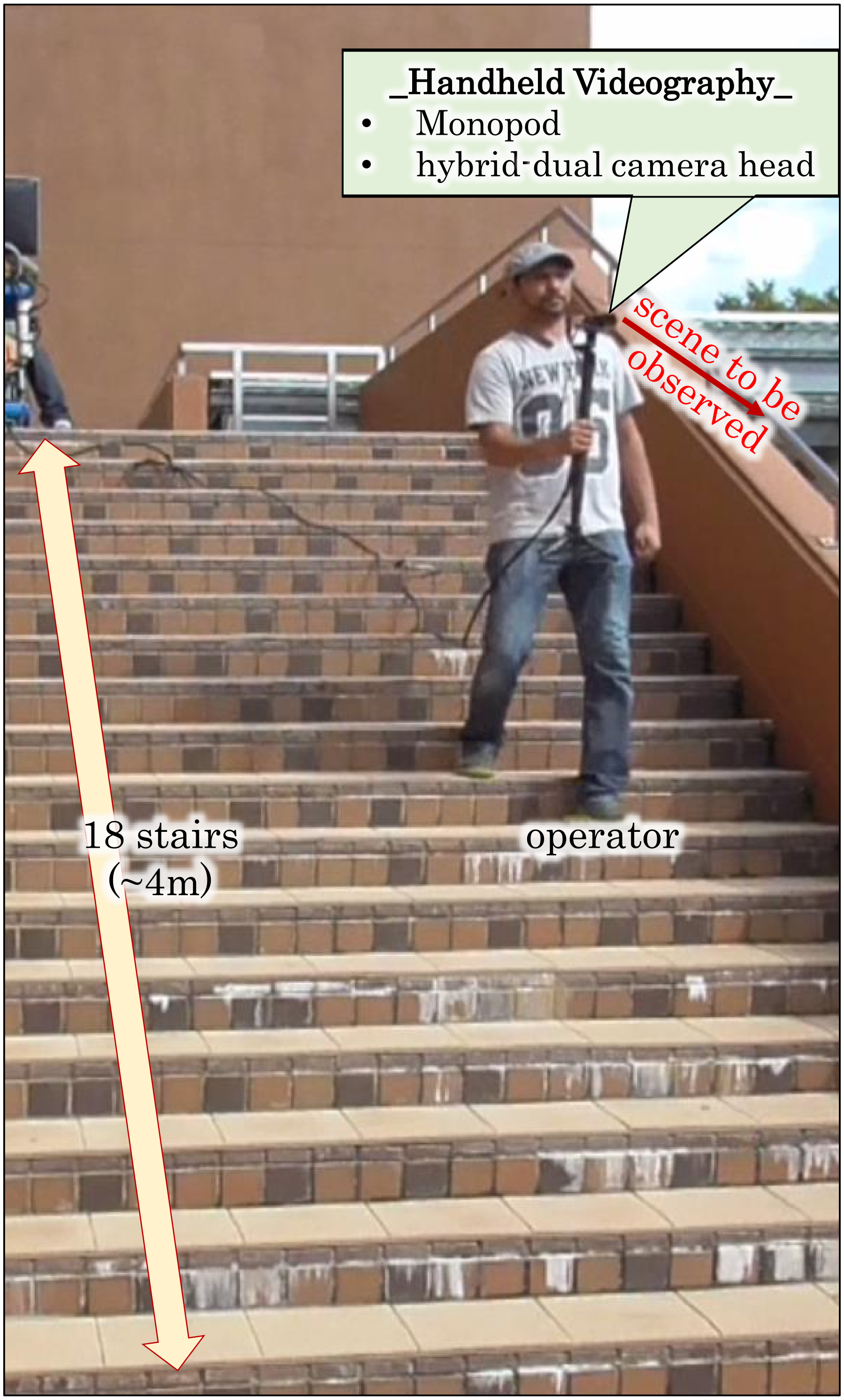
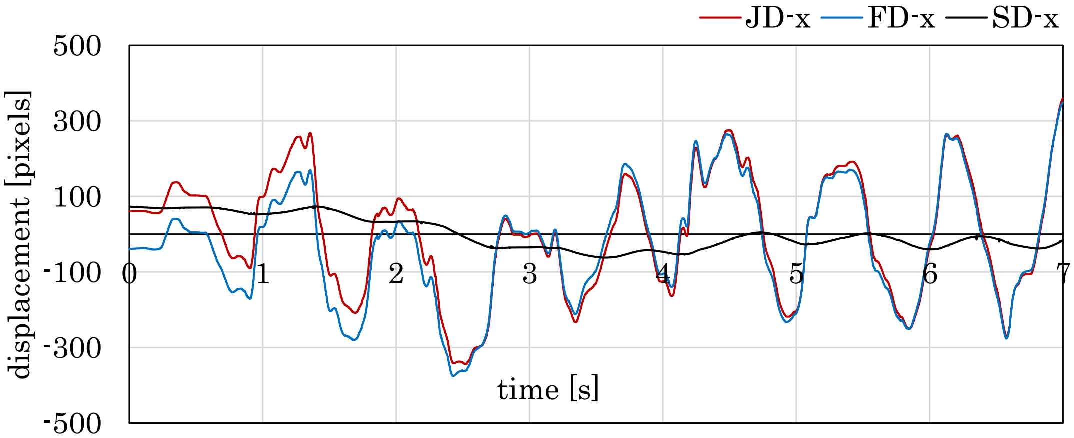
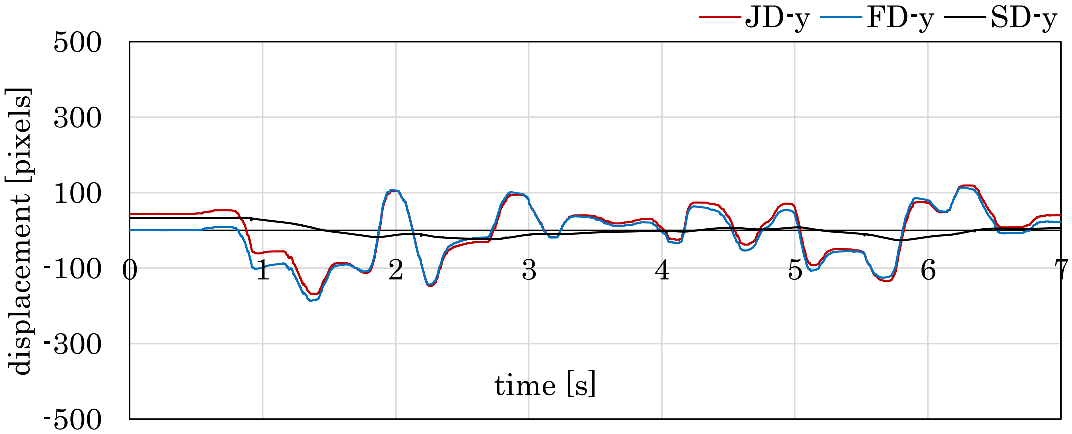


Figure 5.19: Experimental Setup for Hybrid-dual Camera Module

5.2.3.2 Stabilization High-resolution Image Sequences



(a) displacements in x-axis

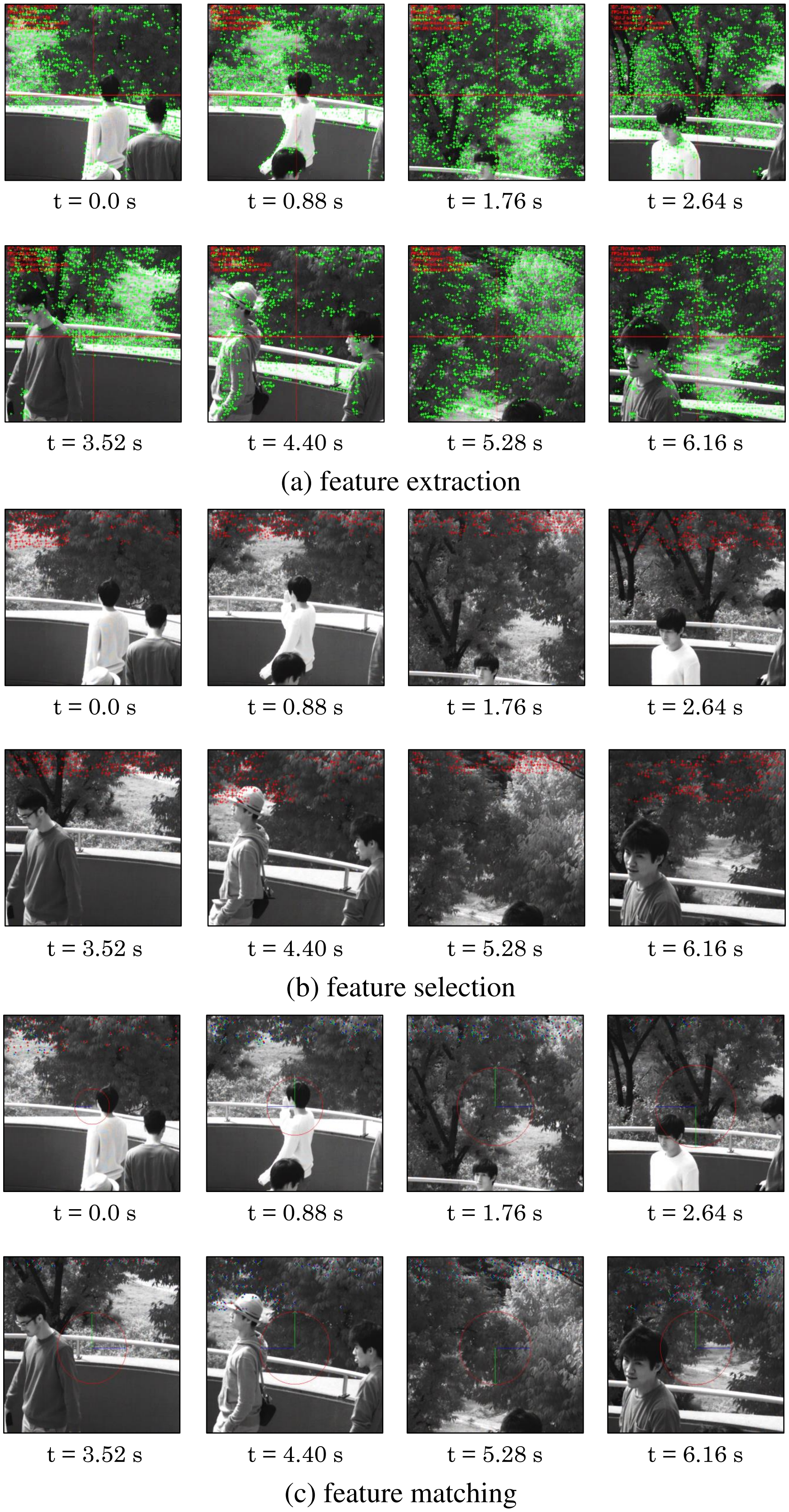


(b) displacements in y-axis

Figure 5.20: Graph Showing Stabilization of Jitter Displacements Conducted using Handheld Hybrid-dual Camera Module

Figure 5.20 shows the JDs, the HDs, and the SDs in (a) the vertical direction and (b) the horizontal directionin the camera 1 image for t = 0-7 s. Figure 5.21 shows (a) the extracted feature points and (b) the pairs of matched feature points, which are plotted in the 512×496 input images of camera 1. Figure 5.22 shows (a) the 2048×2048 input images and (b) the stabilized images of camera 2. Additionally, 2048×2048 images are stabilized in real time at an interval of 12.41 ms; the fastest rate of our stabilization is

80.6 fps. The images in Figures 5.21 and 5.22 were taken for t = 0–6.16 s with an interval of 0.88 s were used to monitor whether the camera ego-motion at approximately 1 Hz was reduced in the stabilized images. According to raster scanning from the upper left to the lower right in the camera 1 image, feature points in its upper region were selected for feature point matching when their number was much larger than 300. Thus, as illustrated in Figure 5.21(b), only 300 feature points located on the background trees in the upper region of the camera 1 image were selected for feature point matching in all the frames and those around walking persons in the center and lower regions were ignored. Video stabilization was conducted based on the static background trees, ignoring the dynamically changing appearances of the walking persons in the center and lower regions of the camera 1 image. In Figure 5.20, the JDs in both the horizontal and vertical directions time-varied at approximately 1 Hz, corresponding to the frequency of the camera ego-motion, which was determined by the relative geometrical relationship between the dual-camera head and the static background trees.It can be observed that the SDs were obtained as the low-frequency component by reducing the high-frequency jitter component and that the 2048×2048 images were stabilized to significantly reduce the apparent motion of the background objects such as trees and a handrail of stairs in the images, as illustrated in Figure 5.22(b).

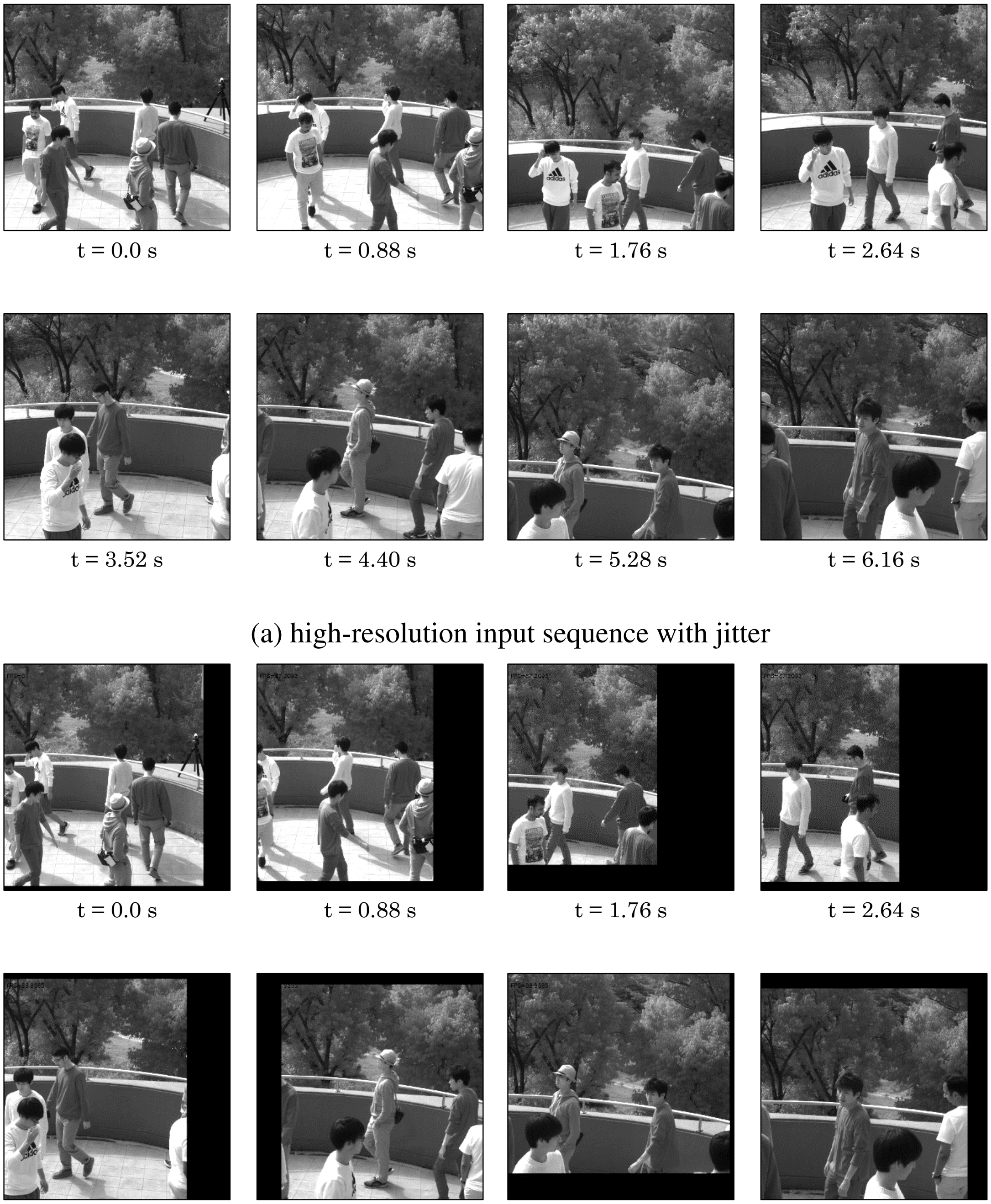


(a) feature extraction

(b) feature selection

(c) feature matching

Figure 5.21: HFR Vision-based Jitter Sensing Conducted using Handheld Hybrid-dual Camera Module



(a) high-resolution input sequence with jitter

(b) high-resolution stabilized sequence with reduced jitter

Figure 5.22: Video Stabilization of High-resolution Jittered Sequence Conducted using Handheld Hybrid-dual Camera Module

We confirmed that the camera jitter with the operator's quick hand motion and the 1 Hz camera jitter in the experiment were correctly measured as the background objects were always observed, with naked eyes in real time, as semi-stationary objects in the stabilized images when they were displayed on a computer display. By selecting the feature points in the static background for feature point matching. These experimental results indicate that our hybrid-camera system can correctly stabilize 2048×2048 images in real time without disturbance from the dynamically changing appearances around the walking persons when assisted by feature-point-based HFR-jitter sensing at 1000 fps even when a walking operator moves the hand-held dual-camera head of our systemquickly. Here, the frequency of the camera jitter may increase depending on the operator's motion, however, our system is capable of stabilizing frequencies much higher than 1 Hz. The operator's motion in the frequency range from 0.5 to 10 Hz can be compensated for video stabilization on our system.

5.2.4 Discussions and Comparisons

5.2.4.1 Discussions

In this study, we introduced a concept of real-time digital video stabilization with HFR video processing, in which an HFR vision system can simultaneously estimate apparent translational motion in image sequences as an HFR jitter sensor and is hybridized to assist for compensating high-resolution image sequences. We developed a hybrid-camera system for real-time high-resolution video stabilization that can simultaneously stabilize 2048×2048 images captured at 80 fps by executing frame-by-frame feature point tracking in real-time at 1000 fps on a 512×512 HFR vision system. The experimental results demonstrated its performance for several moving scenes.

5.2.4.2 Computational Efficiency

	time
(1) Feature point detection	0.07
(2) Feature point matching	0.42
(3) Jitter sensing	0.03
(4) Jitter-compensated Image Composition	12.41
Total (1)–(3)	0.52

Table 5.3: Execution Times on Hybrid-dual Camera System (unit:ms).

Table 5.3 summarizes the execution times of steps (1)–(4) when our algorithm was implemented on the hybrid-camera system with the parameters stated above. The execution time of step (1) includes the image acquisition time for a 512×512 image on the FPGA board of the high-speed vision system. The total execution time of steps (1)–(3) was less than the frame cycle time of the high-speed vision system, Δt = 1 ms. Due to the higher synthesizing cost for 2048×2048 image sequences, the execution time of step (4) was much larger than that of the other steps, but it was less than the frame cycle time of the high-resolution camera $\Delta t'$ = 12.5 ms. Here steps (2)–(4) were software-implemented as multithreaded processes to achieve real-time jitter sensing at 1000 fps in parallel with real-time composition of jitter-compensated high-resolution images at 80 fps to simultaneously display it on a computer display.

5.2.4.3 Compare with Conventional Methods

We compared our algorithm with conventional methods for feature-based video stabilization using SURF [25], SIFT [111], FAST [112], and Harris corner [113], which are distributed in the OpenCV standard library [115]. Table 5.4 shows the execution times for step (1) and steps (2)–(4) when conventional methods for 512×496 and 2048×2048 images on the same PC as that used in our hybrid-camera system. These methods involved the processes for steps (2)–(4) such as descriptor matching, affine transformation for displacement estimation, Kalman filtering for jitter removal, and stabilized image composition. In the evaluation, we assume that the number of feature points to be selected in step (1) are less than $M = 300$ in both the cases of 512×496 and 2048×2048 images. As shown in Table 5.4, the computational cost for synthesizing 2048×2048 images is expressively higher than that for 512×496 images.

methods	512 × 496		2048 × 2048	
	step (1)	steps (2)-(4)	step (1)	steps (2)-(4)
SURF Features [112]	84.03	20.47	742.64	107.59
SIFT Features [111]	147.85	12.44	2194.79	131.19
FAST Features [25]	0.56	11.44	3.19	88.91
Harris Features (SW) [113]	19.56	18.41	219.33	174.82
Harris Features (HW) [our method]	0.07	0.45	—	12.41

Table 5.4: Comparison in Execution Time (unit: ms).

Our algorithm can accelerate the execution time of steps (2)–(4) for video stabilization of 2048×2048 images to 12.41 ms by hybridizing the hardware-implemented feature extraction of 512×496 images in step (1). We confirmed that our method could sense the jitter of several HFR videos, in which frame-by-frame image displacements are small, at the same accuracy level as those of conventional methods. The latter involves a matching process with predictions such as the Kalman filter to compensate specific image displacements between frames in a standard video at dozens of frames per second. Similarly, with the feature point extraction process, such a matching process with prediction is time-consuming to the extent that conventional methods cannot be executed for the real-time video stabilization of 2048×2048 images at dozens of frames per second. Thus, our hybridized algorithm for video stabilization of high-resolution images has computational advantages over conventional feature-based stabilization methods.

Chapter 6

Conclusion

In this study, we emphasized on accelerating displacement and jitter sensing to compose a real-time mosaicked panorama and a smooth video sequence by using an HFR vision-based sensing system. We studied various sensing methods, among them, the feature matching-based approach is used for sensing frame-by-frame displacements in the HFR sequences. The single panoramic images and the smooth videos are synthesized by estimating displacements as image transformation parameters, which is accelerated by implementing a hardware logic of Harris Corner detection on the high-speed vision platform. Further acceleration is conducted by narrowing the feature search range in the consecutive frames considering the redundantly extracted feature points in the HFR sequences. However, the limitation of image-space resolution in the HFR vision systems while composing a stabilized video is overcome by the hybrid camera module. The high-speed camera of 512x512 pixels-size operating at 1000 fps and the high-resolution camera of 2048x2048 pixels-size running at 80 fps, as a hybrid-dual-camera module are placed side-by-side on the same platform with overlapping viewing angle to realize the identical motion anomalies as a jitter. The performance of the Butterworth 5th order IIR filter is also verified and confirmed for the stabilization as a high pass filter. It eliminates the low frequency-high amplitude undesired motion to preserve the high-frequency components in the sequences. The real-time video stabilization and the mosaicked panorama are demonstrated with several confirmations and real-world experiments. The real scenes in which a hybrid camera system had significant jitter displacements due to unsteady egomotion in the handheld operation were efficiently stabilized. We also confirmed the robustness of our implemented system by verifying its performance when the mixed motion was generated by the pan-tilt actuator as a concoction of 1Hz oscillation along with the constant velocity from left to right. In the case of mosaicked panorama, the performance and the stitching path were confirmed by moving an HFR camera head using a pan-tilt actuator, over the indoor scene as well as handheld free-style motion over the real-world outdoor scene. Based on the above-mentioned confirmations and results, we plan to improve our mosaicked panorama system for more robust and long-term image mosaicking for complicated three-dimensional scenes, SLAM, and other surveillance technologies in the real world. In the future, we also plan to investigate in automatic optimization of the parameter for different scenes. In the case of our video stabilization, we aim to improve it for further robust usage with the compensation in 3-D translational and rotational anomalies under time-varying illumination while capturing complicated scenes. Also, apply object recognition and motion segmentation methods to segregate the camera motion by intelligently ignoring feature points around the moving objects. In the future, we extend it to create embedded and consumer camera systems for mobile robots, UAVs, and video stabilizer for a variety of applications.

Bibliography

[1] R. Szeliskki, "Image alignment and stitching: A tutorial," *Tech. Rep.*, MSR-TR-2004-92, Microsoft Research, 2006.

[2] B. Zitova and J. Flusser, "Image registration methods: A survey," *Image Vis. Comput.*, Vol. 21, pp. 977–1000, 2003.

[3] R. Szeliski and S. Kang, "Direct methods for visual scene reconstruction," *Proc. IEEE Workshop on Representations of Visual Scenes*, pp. 26–33, 1995.

[4] H.S. Sawhney and R. Kumar, "True multi-image alignment and its application to mosaicing and lens distortion correction," *IEEE Trans. Patt. Anal. Mach. Intelli.*, Vol. 21, No. 3, pp. 235–243, 1999.

[5] H.-Y. Shum and R. Szeliski, "Systems and experiment paper: Construction of panoramic image Mosaics with global and local alignment," *Int. J. Comput. Vis.*, Vol. 36, No. 2, pp. 101–130, 2000.

[6] I. Zoghlami, O. Faugeras, and R. Deriche, "Using geometric corners to build a 2D mosaic from a set of images," *Proc. Int. Conf. Comput. Vis. Patt. Recog.*, pp. 420–425, 1997.

[7] P. McLauchlan and A. Jaenicke, "Image mosaicing using sequential bundle adjustment," *Image Vis. Comput.*, Vol. 20, No. 9–10, pp. 751–759, 2002.

[8] D. Capel and A. Zisserman, "Computer vision applied to super resolution," *IEEE Signal Processing Magazine*, Vol. 20, No. 3, pp. 75–86, 2003.

[9] M. Brown and D.G. Lowe, “Automatic panoramic image stitching using invariant features,” *Int. J. Comput. Vis.*, Vol. 74, No. 1, pp. 59–73, 2007.

[10] J. Shi and C. Tomasi, “Good features to track,” *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, pp. 593-600, 1994.

[11] D.G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vis.*, Vol. 60, No. 2, pp. 91–110, 2004.

[12] M. Irani, P. Anandan, J.R. Bergen, R. Kumar, and S. Hsu, “Efficient representations of video sequences and their applications,” *Image Commu.*, Vol. 8, No. 4, pp. 327–351, 1996.

[13] A. Bevilacqua and P. Azzari, “High-quality real time motion detection using ptz cameras,” *Proc. IEEE Int. Conf. Video and Signal Based Surveillance*, pp. 23, 2006.

[14] A. Kelly, “Mobile robot localization from large-scale appearance mosaics,” *Int. J. Robotic Res.*. Vol. 19, No. 11, pp. 1104–1125, 2000.

[15] P. Azzari, L. Di Stefano, F. Tombari, and S. Mattoccia, “Markerless augmented reality using image mosaics,” *ICISP 2008. LNCS*, vol. 5099. Springer, Heidelberg, 2008.

[16] M. Kourogi, T. Kurata, J. Hoshino, and Y. Muraoka, “Real-time image mosaicing from a video sequence,” *Proc. Int. Conf. Image Proc.*, pp. 133–137, 1999.

[17] J. Civera, A.J. Davison, J.A. Magallon, and J.M.M. Montiel, “Drift-free real-time sequential mosaicing,” *Int. J. Comput. Vis.*, Vol. 81, pp. 128–137, 2009.

[18] T. Botterill, S. Mills, and R. Green, “Real-time aerial image mosaicing,” *Proc. Int. Conf. Image Vis. Comput. New Zealand*, pp. 1–8, 2010.

[19] R.H.C. de Souza, M. Okutomi, and A. Torii, “Real-time image mosaicing using non-rigid registration,” *Proc. 5th Pacific Rim Conf. on Advances in Image and Video Technology*, pp. 311–322, 2011.

[20] T. Kuwa, Y. Watanabe, T. Komuro, and M. Ishikawa: “Wide range image sensing using a thrown-up camera,” *Proc. IEEE Int. Conf. Multimedia & Expo*, pp. 878-883. 2010.

[21] C. Morimoto and R. Chellappa, “Fast electronic digital image stabilization,” *IEEE Proc. ICPR*, 3, pp. 284–288, 1996.

[22] W. Scott and R. Sergio, “Introduction to Image Stabilization,” *SPIE Press*, 2006, doi: 10.1117/3.685011

[23] J. Yang, D. Schonfeld, and M. Mohamed, “Robust video stabilization based on particle filter tracking of projected camera motion,” *IEEE Trans. Circuits Syst Video Technol.*, 19, 7, pp. 945–954, 2009.

[24] A. Amanatiadis, A. Gasteratos, S. Papadakis, V. Kaburlasos, and A. Ude, “Image Stabilization,” *ARVRV*, pp. 261–274, 2010.

[25] J. Xu, H. W. Chang, S. Yang, and D. Wang, “Fast feature-based video stabilization without accumulative global motion estimation,” *IEEE Trans. Consum. Electron.*, 58, 3, pp. 993–999, 2012.

[26] H. Kusaka, Y. Tsuchida, and T. Shimohata, “Control technology for optical image stabilization,” *SMPTE Motion Imag. J.*, 111, pp. 609–615, 2012.

[27] B. Cardani, “Optical image stabilization for digital cameras,” *IEEE Control Syst.*, 26, pp. 21–22, 2006.

[28] K. Sato, S. Ishizuka, A. Nikami, and M. Sato, “Control techniques for optical image stabilizing system,” *IEEE Trans. Consum. Electron.*, 39, pp. 461–466, 1993.

[29] P. Pournazari, R. Nagamune, and M. A. Chiao, “Concept of a magnetically actuated optical image stabilizer for mobile applications,” *IEEE Trans. Consum. Electron.*, 60, pp. 10–17, 2014.

[30] Q. Hao, X. Cheng, J. Kang, and Y. Jiang, “An image stabilization optical system using deformable freeform mirrors,” *Sensors*, 15, pp. 1736–1749, 2015.

[31] C. W. Chiu, P. C. P. Chao, and D. Y. Wu, “Optimal design of magnetically actuated optical image stabilizer mechanism for cameras in mobile phones via genetic algorithm,” *IEEE Trans. Magn.*, 43, pp. 2582–2584, 2007.

[32] J. Moon and S. Jung, “Implementation of an image stabilization system for a small digital camera,” *IEEE Trans. Consum. Electron.*, 54, pp. 206–212, 2008.

[33] M. Song, Y. Hur, N. Park, K. Park, Y. Park, S. Lim, and J. Park, “Design of a voice-coil actuator for optical image stabilization based on genetic algorithm,” *IEEE Trans. Magn.*, 45, pp. 4558–4561, 2009.

[34] M. Song, H. Baek, N. Park, K. Park, T. Yoon, Y. Park, and S. Lim, “Development of small sized actuator with compliant mechanism for optical image stabilization,” *IEEE Trans. Magn.*, 46, pp. 2369–2372, 2010.

[35] T. S. Li, C. Chen, and Y. Su, “Optical image stabilizing system using fuzzy sliding-mode controller for digital cameras,” *IEEE Trans. Consum. Electron.*, 58, 2, pp. 237–245, 2012, doi:10.1109/TCE.2012.6227418.

[36] C. D. Walrath, “Adaptive bearing friction compensation based on recent knowledge of dynamic friction,” *Automatica*, 20, pp. 717–727, 1984.

[37] B. Ekstrand, “Equations of motion for a two-axes gimbal system,” *IEEE Trans. Aerosp. Electron. Syst.*, 37, pp. 1083–1091, 2001.

[38] P. J. Kennedy and R. L. Kennedy, “Direct versus indirect line of sight (LOS) stabilization,” *IEEE Trans. Control Syst. Technol.*, 11, pp. 3–15, 2003.

[39] X. Zhou, Y. Jia, Q. Zhao, and R. Yu, Experimental validation of a compound control scheme for a two-axis inertially stabilized platform with multi-Sensors in an unmanned helicopter-based airborne power line inspection system,” *Sensors*, 16, 2016, doi:10.3390/s16030366.

[40] S. W. Jang, M. Pomplun, G. Y. Kim, and H. I. Choi, “Adaptive robust estimation of affine parameters from block motion vectors,” *Image Vis. Comput.*, 23, pp. 1250–1263, 2005.

[41] L. Xu and X. Lin, “Digital image stabilization based on circular block matching,” *IEEE Trans. Consum. Electron.*, 52, 2, pp. 566–574, 2006, doi:10.1109/TCE.2006.1649681.

[42] Y. Moshe and H. Hel-Or, “Video block motion estimation based on gray-code kernel,” *IEEE Trans. Image Process.*, 18, 10, pp. 2243–2254, 2009, doi: 10.1109/TIP.2009.2025559

[43] W. Chantara, J. H. Mun, D. W. Shin, and Y. S. Ho, “Object tracking using adaptive template matching,” *IEIE SPC*, 4, pp. 1–9, 2015.

[44] S. Ko, S. Lee, and K. Lee, “Digital image stabilizing algorithms based on bit-plane matching,” *IEEE Trans. Consum. Electron.*, 44, pp. 617–622, 1998.

[45] S. Ko, S. Lee, S. Jeon, and E. Kang, “Fast digital image stabilizer based on Gray-coded bit-plane matching” *IEEE Trans. Consum. Electron.*, 45, pp. 598–603, 1999.

[46] A. Litvin, J. Konrad, and W. C. Karl, “Probabilistic video stabilization using Kalman filtering and mosaicking,” *Proc. SPIE 5022, Image and Video Commun. Process.*, pp. 20–24, 2003, doi:10.1117/12.476436

[47] K. K. Rasheed, T. Zafar, S. Mathavan, and M. Rahman, “Stabilization of 3D pavement images for pothole metrology using the Kalman filter,” *IEEE 18th Int. Conf. Intell. Transport. Syst.*, pp. 2671–2676, 2015.

[48] S. Erturk, “Image sequence stabilisation based on Kalman filtering of frame positions,” *Electron. Lett.*, 37, 20, pp. 1217–1219, 2001.

[49] S. Erturk, “Real-time digital image stabilization using Kalman filters,” *J. Real-Time Imag.*, 8, 4, pp. 317–328, 2002.

[50] C. Wang, J. H. Kim, K. Y. Byun, J. Ni, and S. J. Ko, “Robust digital image stabilization using the Kalman filter,” *IEEE Trans. Consum. Electron.*, 55, 1, pp. 6-14, 2009, doi: 10.1109/TCE.2009.4814407

[51] S. Erturk and T. J. Dennis, “Image sequence stabilisation based on DFT filtering,” *IEEE Proc. Vis. Imag. Sig. Process.*, 147, 2, pp. 95–102, 2000.

[52] Y. Junlan, D. Schonfeld, and M. Mohamed, “Robust video stabilization based on particle filter tracking of projected camera motion,” *IEEE Trans. Circuits Syst. Video Technol.*, 19, 7, pp. 945–954, 2009.

[53] S. Hong and E. Atkins, “Moving sensor video image processing enhanced with elimination of ego motion by global registration and SIFT,” *IEEE Int. Tools Artif. Intell.*, pp. 37–40, 2008.

[54] R. Hu, R. Shi, I. F. Shen, and W. Chen, “Video stabilization using scale-invariant features,”*11th Int. Conf.,Zurich Info. Visual.*, pp. 871–877, 2007.

[55] Y. Shen, P. Guturu, T. Damarla, B. P. Buckles, and K. R. Namuduri, “Video stabilization using principal component analysis and scale invariant feature transform in particle filter framework, ”*IEEE Trans. Consum. Electron.*, 55, pp. 1714–1721, 2009.

[56] S. Liu, L. Yuan, P. Tan, and J. Sun, “Bundled camera paths for video stabilization,”*ACM Trans. Graphics*, 32, 4, 78, pp. 1–10, 2013, doi:10.1145/2461912.2461995.

[57] S. K. Kim, S. J. Kang, T. S. Wang, and S. J. Ko, “Feature point classification based global motion estimation for video stabilization,”*IEEE Trans. Consum. Electron.*, 59, pp. 267–272, 2013.

[58] X. Cheng, Q. Hao, and M. Xie, “A comprehensive motion estimation technique for the improvement of EIS methods based on the SURF algorithm and Kalman filter,” *Sensors*, 16, 2016, doi:10.3390/s16040486.

[59] S. Jeon, I. Yoon, J. Jang, S. Yang, J. Kim, and J. Paik, “Robust video stabilization using particle keypoint update and l1-optimized camera path,” *Sensors*, 17, 2017, doi:10.3390/s17020337.

[60] J. Chang, W. Hu, M. Cheng, and B. Chang, “Digital image translational and rotational motion stabilization using optical ow technique,” *IEEE Trans. Consum. Electron.*, 48, pp. 108–115, 2002.

[61] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H. Y. Shum, “Full-frame video stabilization with motion inpainting,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 28, pp. 1150–1163, 2006.

[62] J. Cai and R. Walker, “Robust video stabilization algorithm using feature point selection and delta optical flow,” *IET Comput. Vis.* , 3, 4, pp. 176–188, 2009.

[63] N. Ejaz, W. Kim, S. I. Kwon, and S. W. Baik, “Video stabilization by detecting intentional and unintentional camera motions,” *Third Inter. Conf. Intel. Sys. Model. Sim.*, pp. 312–316, 2012.

[64] W. Xu, X. Lai, D. Xu, and N. A. Tsoligkas, “An integrated new scheme for digital video stabilization,”*Adv. Multimed.*, 8, 2013, doi:10.1155/2013/651650.

[65] S. Liu, L. Yuan, P. Tan, and J. Sun, “Steady flow: Spatially smooth optical flow for video stabilization,”*IEEE Conf. omput. Vis. Pattern Recog.*, pp. 4209–4216, 2014.

[66] W. Lu, Z. Hongying, G. Shiyi, M. Ying, and L. Sijie, “The adaptive compensation algorithm for small UAV image stabilization,”*IEEE Inter. Geo. Remote Sens. Sym.*, pp. 4391–4394, 2012.

[67] K. Mayen, C. Espinoza, H. Romero, S. Salazar, M. Lizarraga, and R. Lozano, “Real-time video stabilization algorithm based on efficient block matching for UAVs,” *Works. Resea., Educ. Develop. Unman. Aerial Syst.*, pp. 78–83, 2015.

[68] S. Hong, T. Hong, and Y. Wu, “Multi-resolution unmanned aerial vehicle video stabilization,”*Proc. IEEE Nat. Aero. Elect. Conf.*, 14,16, pp. 126–131, 2010.

[69] P. Y. Oh and W. E. Green, “Mechatronic kite and camera rig to rapidly acquire, process, and distribute aerial images,” *IEEE/ASME Trans. Mech.*, 9, 4, pp. 671–678, 2004.

[70] M. Ramachandran and R. Chellappa, “Stabilization and mosaicing of airborne videos,” *Inter. Conf. Imag. Process.*, pp. 345–348, 2006.

[71] M. Ax, S. Thamke, L. Kuhnert, J. Schlemper, and K. D. Kuhnert, “Optical position stabilization of an UAV for autonomous landing,” *ROBOTIK 7th German Conf. Robo.*, pp. 1–6, 2012.

[72] W. Ahlem, W. Ali, and M. A. Adel, “Video stabilization for aerial video surveillance,”*AASRI Procedia*, 4, pp. 72–77, 2013.

[73] C. Morimoto and R. Chellappa, “Fast electronic digital image stabilization for off-road navigation,” *J. Real-Time Imag.* , 2, 5, pp. 285–296, 1996.

[74] Y. S. Yao and R. Chellapa, “Selective stabilization of images acquired by unmanned ground vehicles,” *IEEE Trans. Robo. Autom.*, 13, 5, pp. 693–708, 1997.

[75] G. L. Foresti, “Object recognition and tracking for remote video surveillance,” *IEEE Trans. Circuits Syst. Video Technol.*, pp. 1045–1062, 1999.

[76] A. Ferreira and J. G. Fontaine, “Coarse/fine motion control of a teleoperated autonomous piezoelectric nanopositioner operating under a microscope,” *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mech.*, 2, pp. 1313–1318, 2001.

[77] J. Zhu, C. Li, and J. Xu, “Digital image stabilization for cameras on moving platform,” *Int. Conf. Intell. Info. Hid. Mult. Sig. Process.*, pp. 255–258, 2015.

[78] C. Guestrin, F. Cozman, and S. M. Godoy, “Industrial applications of image mosaicing and stabilization,” *1998 Sec. Int. Conf. Knowledge-Based Intell. Electron. Syst.*, 2, pp. 174–183, 1998, doi: 10.1109/KES.1998.725908

[79] J. Lobo, J. F. Ferreira, and J. Dias, “Robotic implementation of biological bayesian models towards visuo-inertial image stabilization and gaze control,” *IEEE Int. Conf. Robo. Biomed.*, pp. 443–448, 2009.

[80] B. M. Smith, L. Zhang, H. Jin, and A. Agarwala, “Light field video stabilization,” *IEEE 12th Int. Conf. Comp. Vis.*, pp. 341–348, 2009.

[81] Z. Li, S. Pundlik, and G. Luo, “Stabilization of magnified videos on a mobile device for visually impaired,”*IEEE Conf. Comput. Vis. Pattern Recog. Works.*, pp. 54–55, 2013.

[82] A. Roncone, U. Pattacini, G. Metta, and L. Natale, “Gaze stabilization for humanoid robots: A comprehensive framework,” *IEEE-RAS Int. Conf. Human. Robo.*, pp. 259–264, 2014.

[83] M. Hansen, P. Anandan, K. Dana, G. van der Wal, and P. Burt, “Real-time scene stabilization and mosaic construction,” *Proc. Second IEEE Works. Appl. Comput. Vis.*, 5, 7, pp. 54–62,1994.

[84] S. Battiato, G. Puglisi, and A. R. Bruna, “A robust video stabilization system by adaptive motion vectors filtering,” *IEEE Int. Conf. Mult. Expo*, pp. 373–376, 2008.

[85] M. H. Shakoor and A. R. Dehghani, “Fast digital image stabilization by motion vector prediction,” *The 2nd Int. Conf. Env. Sci. Info. Appl. Technol.*, pp. 151–154, 2010.

[86] L. Araneda and M. Figueroa, “Real-time digital video stabilization on an FPGA,” *2014, 17th Euromicro Conf. Digi. Sys. Desi.*, pp. 90–97, 2014, doi: 10.1109/DSD.2014.26

[87] S. Chang, Y. Zhong, Z. Quan, Y. Hong, J. Zeng, and D. Du, “A real-time object tracking and image stabilization system for photographing in vibration environment using OpenTLD algorithm,” *IEEE Work. Adv. Robot. Social Impact*, pp. 141–145, 2016.

[88] W. Yang, Z. Zhang, Y. Zhang, X. Lu, J. Li, and Z. Shi, “Real-time digital image stabilization based on regional field image gray projection,” *J. Sys. Eng. Electron.*, 27, 1, pp. 224–231, 2016.

[89] J. Dong and H. Liu, “Video stabilization for strict real-time applications,”*IEEE Trans. Circuits Syst. Video Technol.*, 27, 4, pp. 716–724, 2017, doi: 10.1109/TCSVT.2016.2589860

[90] Y. Watanabe, T. Komura, and M. Ishikawa, “955-fps real-time shape measurement of a moving/deforming object using high-speed vision for numerous-point analysis,” *Proc. IEEE Conf. Robot. Autom.*, pp. 3192–3197, 2007.

[91] I. Ishii, T. Taniguchi, R. Sukenobe, and K. Yamamoto, “Development of high-speed and real-time vision platform, H3 vision,”*Proc. IEEE Conf. Intell. Robots Syst.*, pp. 3671–3678, 2009.

[92] I. Ishii, T. Tatebe, Q. Gu, Y. Moriue, T. Takaki, and K. Tajima, “2000 fps real-time vision system with high-frame-rate video recording,” *Proc. IEEE Conf. Robot. Autom.*, pp. 1536–1541, 2010.

[93] T. Yamazaki, H. Katayama, S. Uehara, A. Nose, M. Kobayashi, S. Shida, M. Odahara, K. Takamiya, Y. Hisamatsu, S. Matsumoto, L. Miyashita, Y. Watanabe, T. Izawa, Y. Muramatsu, and M. Ishikawa, “A 1 ms high-speed vision chip with 3D-stacked 140GOPS column-parallel PEs for spatio-temporal image processing,” *Proc. Conf. Solid-State Circuits*, pp. 82–83, 2017.

[94] A. Namiki, K. Hashimoto, and M. Ishikawa, “Hierarchical control architecture for high-speed visual servoing,” *IJRR*, 22, 873–888, 2003.

[95] T. Senoo, A. Namiki, and M. Ishikawa, “Ball control in high-speed batting motion using hybrid trajectory generator,” *Proc. IEEE Conf. Robot. Autom.*, pp. 1762–1767, 2006.

[96] A. Namiki and N. Ito, "Ball catching in kendama game by estimating grasp conditions based on a high-speed vision system and tactile sensors," *Proc. IEEE Conf. Human. Robots*, pp. 634–639, 2014.

[97] T. Aoyama, T. Takaki, T. Miura, Q. Gu and I. Ishii, "Realization of flower stick rotation using robotic arm," *Proc. IEEE Conf. Intell. Robots Syst.*, pp. 5648–5653, 2015.

[98] M. Jiang, T. Aoyama, T. Takaki, and I. Ishii, "Pixel-level and robust vibration source sensing in high-frame-rate video analysis," *Sensors*, 16, 2016, doi:10.3390/s16111842.

[99] M. Jiang, Q. Gu, T. Aoyama, T. Takaki, and I. Ishii, "Real-Time Vibration Source Tracking using High-Speed Vision," *IEEE Sensors J.*, 17, pp. 1513–1527, 2017.

[100] Y. Watanabe, T. Komuro, and M. Ishikawa, "955-fps real-time shape measurement of a moving/deforming object using high-speed vision for numerous-point analysis," *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 3192–3197, 2007.

[101] S. Hirai, M. Zakoji, A. Masubuchi, and T. Tsuboi, "Realtime FPGA-based vision system," *J. Robot. Mechat.*, Vol. 17, pp. 401–409, 2005.

[102] I. Ishii, R. Sukenobe, T. Taniguchi, and K. Yamamoto, "Development of high-speed and real-time vision platform, H^3 Vision," *Proc. IEEE/RSJ Int. Conf. Intelli. Rob. Sys.*, pp. 3671–3678, 2009.

[103] I. Ishii, T. Tatebe, Q. Gu, Y. Moriue, T. Takaki, and K. Tajima, "2000 fps real-time vision system with high-frame-rate video recording," *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 1536–1541, 2010.

[104] Q. Gu, T. Takaki, and I. Ishii, "Fast FPGA-based multi-object feature extraction," *IEEE Trans. Circ. Sys. Video Tech.*, Vol. 23, No. 1, pp.30-45, 2013.

[105] I. Ishii, T. Taniguchi, K. Yamamoto, and T. Takaki, "High-frame-rate optical flow system,"*IEEE Trans. Circu. Syst. Video Technol.*, 22, 1, pp. 105–112, 2012, doi:10.1109/TCSVT.2011.2158340

[106] I. Ishii, T. Tatebe, Q. Gu, and T. Takaki, "Color-Histogram-Based Tracking at 2000 fps," *J. Electron. Imaging*, 21,1, pp. 1–14, 2012, doi:10.1117/1.JEI.21.1.013010.

[107] Q. Gu, T. Takaki, and I. Ishii, "Fast FPGA-Based Multi-Object Feature Extraction," *IEEE Trans. Circuits Syst. Video Technol.*, 23, pp. 30–45, 2013.

[108] Q. Gu, S. Raut, K. Okumura, T. Aoyama, T. Takaki, and I. Ishii, "Real-Time Image Mosaicing System Using a High-Frame-Rate Video Sequence," *JRM*, 27, pp. 12–23, 2015.

[109] I. Ishii, T. Ichida, Q. Gu, and T. Takaki, "500-fps face tracking system," *JRTIP*, 8, pp. 379–388, 2013.

[110] C. Harris and M. Stephens, "A combined corner and edge detector," *Proc. the 4th Alvey Vis. Conf.*, pp. 147–151, 1988.

[111] S. Battiato, G. Gallo, G. Puglisi, and S. Scellato, "SIFT Features Tracking for Video Stabilization,"*14th Intern. Conf. on Image Analy. and Process.*, pp. 825–830, 2007.

[112] B. Pinto and P. R. Anurenjan, "Video stabilization using Speeded Up Robust Features," *2011 Intern. Conf. Commu. and Sig. Process.*, pp. 527–531, 2011.

[113] A. Lim, B. Ramesh, Y. Yang, C. Xiang, Z. Gao, and F. Lin, "Real-time optical flow-based video stabilization for unmanned aerial vehicles," *J. Real-Time Image Process.*, pp. 1–11, 2017, doi: 10.1007/s11554-017-0699-y

[114] S. Raut, K. Shimasaki, S. Singh, T. Takaki, I. Ishii, "Real-time high-resolution video stabilization using high-frame-rate jitter sensing," *ROBOMECH Journal*, 6, 16, pp 1–21, 2019, doi:10.1186/s40648-019-0144-z

[115] `https://opencv.org/`

Acknowledgment

Firstly, I would like to express my sincere gratitude to my advisor **Prof. Idaku Ishii**, for providing me an opportunity to join the Robotics laboratory, Hiroshima University and the continuous support of my Ph.D. study and related research, as well as for his patience, motivation, and immense knowledge. His advice helped me in all the time of research and writing of this thesis.

Furthermore, I would like to thank the rest of my thesis committee members, **Prof. Toru Yamamoto**, and **Prof. Toshio Tsuji**, for their perceptive comments and inspiration to widen my research from various perspectives. My sincere thanks also go to **Prof. Takeshi Takaki**, **Prof. Tadayoshi Aoyama**, and **Prof. Qingyi Gu**, who provided me an opportunity to join their team, and who gave access to the research facilities. Without their precious support, it would not be possible to conduct this research.

I thank my fellow labmates for friendly discussions related to research. In particular, I am grateful to **Ms. Yukari Kaneyuki** and other secretaries in our laboratory for enlightening me on the glance of academic support.

Last but not least, I would like to thank my family members, and to my parents for encouraging me throughout writing this thesis and my life in general.

March, 2020
Raut Sushil