Received December 7, 2019, accepted December 29, 2019, date of publication January 8, 2020, date of current version January 27, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2964801

Motion Recognition-Based Robot Arm Control System Using Head Mounted Display

YEOHUN YUN, SEUNG JOON LEE, AND SUK-JU KANG, (Member, IEEE)
Department of Electronic Engineering, Sogang University, Seoul 04107, South Korea

Corresponding author: Suk-Ju Kang (sjkang@sogang.ac.kr)

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2020-2018-0-01421) supervised by the IITP (Institute of Information & communications Technology Planning & Evaluation), and Korea Electric Power Corporation (Grant number: R17XA05-28).

ABSTRACT This paper proposes a novel remote monitoring system using a head mounted display (HMD). By using an HMD, a high degree of immersion and the sense of reality is provided to the user. Further, it becomes more convenient to move the camera position while controlling the robot arm on which the camera is mounted through the coordinates obtained from the HMD sensors. The proposed system performs two tasks. Initially, in order to render images in the HMD, the camera captures input images, and the PC connected to the robot arm sends the image to the PC connected to the HMD in real-time after which the image is rendered in the HMD. For the real-time monitoring, we increased the frame per second by reducing the data size. In order to reduce the data size, we define region of interest which is the region a user can see. Then, region of interest of the image is cropped and the resolution of entire image is degraded. Therefore, two images composed of cropped image and degraded image are transmitted and merged into an image. In this way, we can reduce the data size and provide the user with monitoring the image of the original quality. Next, using the inertial measurement unit sensor and base station sensors built into the HMD, the user controls the displacement of the camera during rotation and translation by controlling the robot arm by the user's own motion. The PC connected to the HMD transmits the motion coordinates of the HMD acquired from the sensor into the PC connected to the robot arm. To control the translation of the camera, we define a coordinate system that represents the y-axis as running from top to bottom, and the x-axis as running from the front to the back in three-dimensional space. Thus, each step motor along each axis, which controls the robot arm, has its movement controlled by supplying appropriate values of angular degrees. Through the above process, the proposed method implements a remote monitoring system with five degrees of freedom. In the experiment, we measured the data transmission delay time and the displacement error due to the robot arm's motion control to determine whether the system is suitable for a real-time remote monitoring system. Experimental results show that the proposed system can be optimally operated with a frame rate of 15 frames per second and a group of pixel resolutions of 640×640 of the cropped image and 240×120 of the degraded image. In addition, the average error rate of the robot arm's displacement was 6.45% when the camera position was controlled through the robot arm.

INDEX TERMS Head mounted display, real-time control, remote monitoring system, robot arm.

I. INTRODUCTION

The remote monitoring system enables users to acquire images from the camera at a remote location and monitor the images through a display device. There are various methods that can be used to realize this goal, such as remote monitoring of a fixed area using closed circuit television (CCTV) [1], [2], and remote monitoring by controlling the position of the camera through the use of a robot [3]–[6]. Donato *et al.*[5] proposed a remote monitoring system using an autonomous mobile robot. The autonomous mobile robot is equipped with a monocular camera, a laser scanner, encoders, and a radio frequency identification (RFID) device. By using autonomous mobile

The associate editor coordinating the review of this manuscript and approving it for publication was Yan-Jun Liu.

robots, Donato *et al.*[5] implemented a vision scene change detector and a larger scene change detector to detect unexpected changes such as the addition or removal of objects. Liu *et al.* [6] proposed a remote monitoring system that enables security personnel to track intruders remotely using an internet-based security robot. The internet-based security robot identifies intruders by facial recognition using features such as facial expression, three-dimensional posture, and personal appearance. In addition, remote monitoring systems using drones [7]–[9] have also been studied. However, the drones are difficult to use due to the low battery capacity and a lack of safety.

Remote monitoring systems can be efficiently used in many fields from large and complex facilities such as factories and power plants, to smaller spaces meant for individuals, such as houses. For example, when we consider a surveillance system in a hazardous environment such as a factory or a power plant, the surveillant is not exposed to the environment directly, thereby reducing the potential for harm to the surveillant. In addition, the surveillant can easily monitor the scene without any extra effort at remote sites. Recently, by using various sensors, remote monitoring systems have been developed.

Users may want a realistic monitoring system. For example, when the user experiences a remote location through a remote monitoring system, the user may want to have realistic monitoring feeling like the user is in the monitoring location. However, conventional remote monitoring systems are less stereoscopic and immersive, because they are monitored by displays such as monitors.

With the development of the virtual reality (VR) and augmented reality (AR) industries, the use of head mounted displays (HMDs) has increased, and various products such as Facebook Oculus [10], HTC Vive [11], and Microsoft Hololens [12] have been released. HMD senses the head movement of the user through head tracking, and can adjust the user's viewpoint.

In this study, we proposed a novel monitoring system to provide the sense of reality through an HMD, and to automatically control the camera position according to the movement of the user using sensors in the HMD. The proposed method maximizes the stereoscopic and immersive feeling through the HMD designed with the pupil forming system, partial binocular overlap, and eye tracking system [13], rather than the display used in conventional monitoring systems [1]–[9]. Therefore, it provides a realistic monitoring system to users, and it can improve the immersion experience of the conventional monitoring system. The main contributions of the proposed system are as follows:

- While the traditional method does not consider the realistic monitoring environment, the proposed system focuses on realistic remote monitoring by using sensors of the HMD which provides the stereoscopic effect for monitoring and the realistic control of the robot arm for moving the camera position.
- The proposed system controls the camera position according to the user's movement through the robot arm on which the camera is mounted. In addition, the user's height and robot arm's height are mapped in consideration of the different height for each user to reflect the whole movement of the user's height. Also, we formulated an equation to obtain the proportional ratio that synchronizes the movement of the HMD with the movement of the robot arm. By applying this ratio of movement to control the robot arm, the camera translation control system can adjust the camera position more precisely.
- For real-time image transmission, the proposed system decreases the original image resolution to reduce the data size. However, we define region of interest (RoI) that the user can see. Then, the image of RoI is cropped same as an original resolution and the whole image resolution is decreased. When the image is rendered on the HMD, the cropped image of RoI and the degraded image are merged. Therefore, we can not only reduce the data size, but also provide user to monitor the image with the original resolution.

II. PROPOSED REMOTE MONITORING SYSTEM BASED ON HMD AND ROBOT ARM

Fig. 1 shows the overall proposed remote monitoring system. The proposed system consists of the transmission and rendering of image data and the control of the rotation and translation of the camera (installed on one end of the robotic arm), based on the user's HMD motion, via the use of the robot arm. In this system, the PC connected to the robot arm is used as a client, and the PC connected to the HMD is used as a server. The details are as follows.

A. IMAGE TRANSMISSION AND RENDERING

Fig. 2 shows the image transmission and rendering system. At the start of the image transmission and rendering process, the image transmission system obtains the current frame data. The frame data is captured by a camera module mounted on the robot arm, and it is sent to the client PC. The raw frame data is converted to a byte array before transmission, and it is transmitted to the server PC. When all the data has been transmitted, the server PC converts the byte array back into an image. Finally, the reconstructed image is rendered on the HMD, and the whole process is repeated.

1) IMAGE CAPTURING AND PREPROCESSING OF DATA

The camera module used for the image capture is the RICOH Theta S [14], and it uses USB serial communication. In addition, it has a pixel resolution of 1920×960 and a transmission rate of 15 frames per second (fps). In order to transmit and render an image to the HMD in real-time, rapid image transmission is needed. The frame rate can be increased by reducing the resolution of the original image. Therefore, preprocessing to modify the resolution of the image and

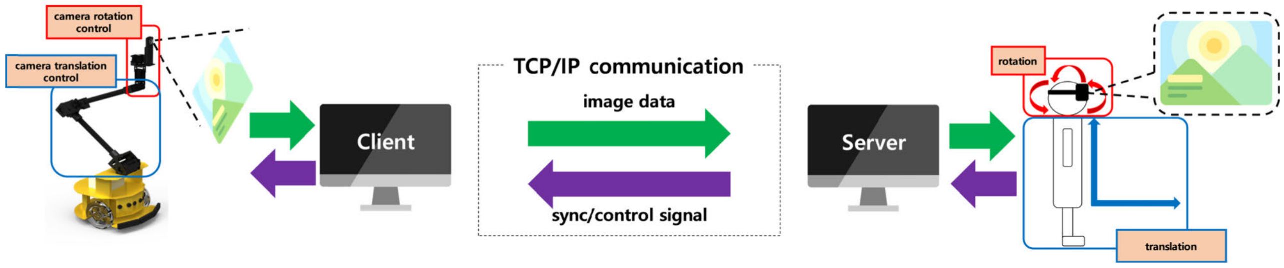


FIGURE 1. Proposed remote monitoring system: This system controls the position of the robot arm with the camera attached to the client PC and the Server PC, and renders the image acquired from the camera to the user's HMD, thus realizing a remote monitoring system.

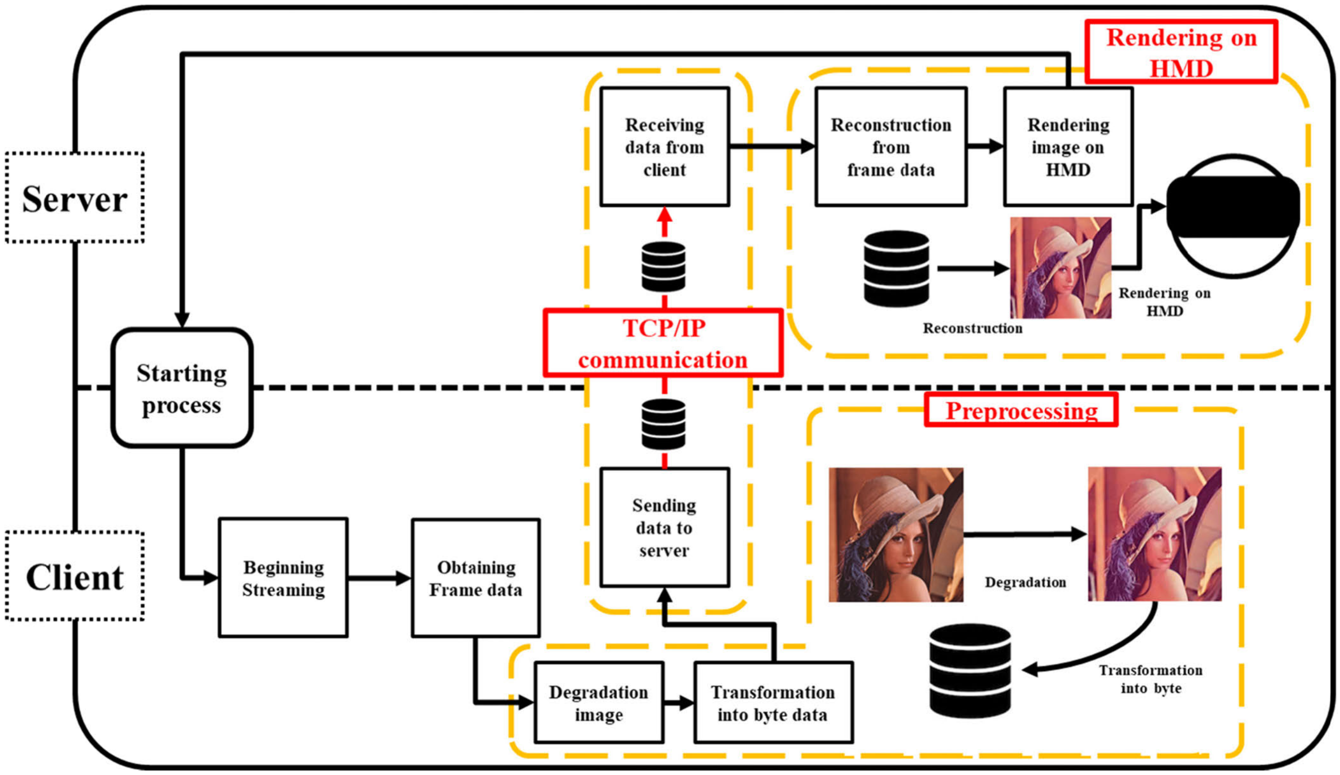


FIGURE 2. Block diagram for image transmission and rendering: The client acquires the image and delivers the image to the server via TCP/IP communication and renders the image on the HMD provided to the user.

its transformation into a byte array is required before the transmission of an acquired image. The captured image is in the bitmap format, and this data type is converted to the joint photographic experts group (JPEG) format [15] stored in the backup storage.

If the frame rate of the original data is less than 15 fps, the resolution of the image is modified and the image is resized. At this time, if the resolution of the image is degraded to reduce the size of the data, the user may monitor the image with a low resolution, and it can cause heterogeneity. Therefore, as shown in Fig. 3, considering that the camera used in the proposed system is an omnidirectional camera with a binocular lens, when the image resolution is degraded, the image resolution needs to be degraded without RoI. In this way, because the camera is controlled by the user's head rotation in real-time, the user can only see the image acquired from the center of the front side lens. Therefore, RoI is defined as a pixel resolution of 640 × 640 which is center of region acquired from front side lens of binocular. While reducing the size, the quality of the image viewed by the user can be provided in the same quality as the original. This is because the video may be cut-off when real-time image rendering is performed, if the frame rate is less than 15 fps. After adjustment of the resolution, the frame data stored in two memory streams is converted into byte arrays. Finally, the frame data stored as byte arrays is ready to be transmitted from the client to the server.

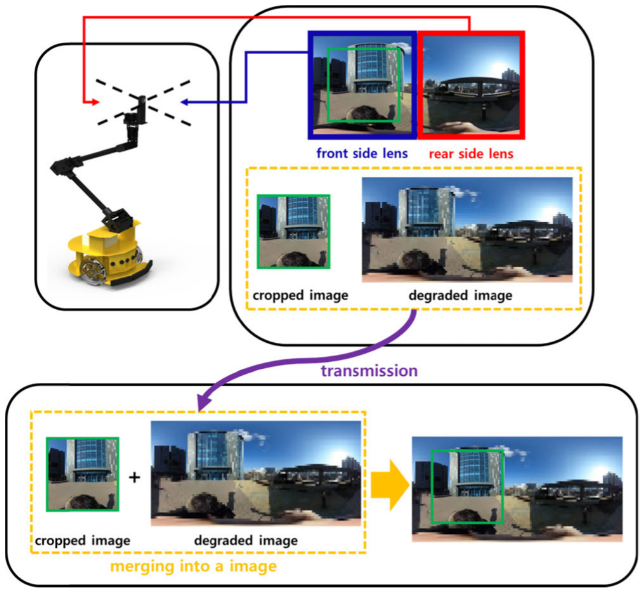


FIGURE 3. Concept for image preprocessing: The image acquired from binocular camera may divide into the region of front side and rear side. And then, the image of visible region is cropped for maintaining original quality and the whole image is degraded for reducing image data size. After images are transmitted, the cropped image and degraded image are merged into an image for rendering.

2) IMAGE TRANSMISSION AND RECEPTION

For a real-time monitoring system using HMD, a stable data transmission network environment is needed. Although various communication techniques exist, the proposed system uses the transmission control protocol/internet protocol (TCP/IP) for its communication [16]. The TCP/IP communication scheme carries out the transmission of the data to the receiver upon receiving the ready signal of the receiver. Further, once the data is fully transmitted, the sender receives a response signal from the receiver; if such a response signal is not received, the sender retransmits the data. Therefore, each step of the data transmission and reception workflow processes a single frame image stored in the byte array format, and the process is repeated until all the frame images are transmitted. In the proposed system, a frame image is divided into packets, each of size 1024 bytes.

3) RENDERING IMAGES TO HMD

While the client continuously transmits frame images to the server, the server needs to render the frame image to the HMD independently during such a transmission. In order to render the frame image on the HMD, the image should be reconstructed with the received byte array data composed of the RoI and entire images. Finally, two images are merged into an image to render on the HMD. Therefore, we implemented the rendering process in Unity 3D, which provides the Steam VR plugin software development kit (SDK) [17] that enables collaboration between the HMD and the development environment. As a result, the real-time image transmission and rendering system is implemented by continuously transmitting the frame data of the image captured by the client to the server and rendering the frame images to the HMD.

B. CONTROLLING THE ROBOT ARM USING HMD

In order to perform remote monitoring, images should be acquired at various rotated and translated positions. Therefore, our proposed system is designed so that the rotation and translation positions of the camera which is mounted on the end of robot arm can be controlled by using a robot arm having five step motors. The motion control of the robot arm can be classified into the rotation control and the translation control of the camera.

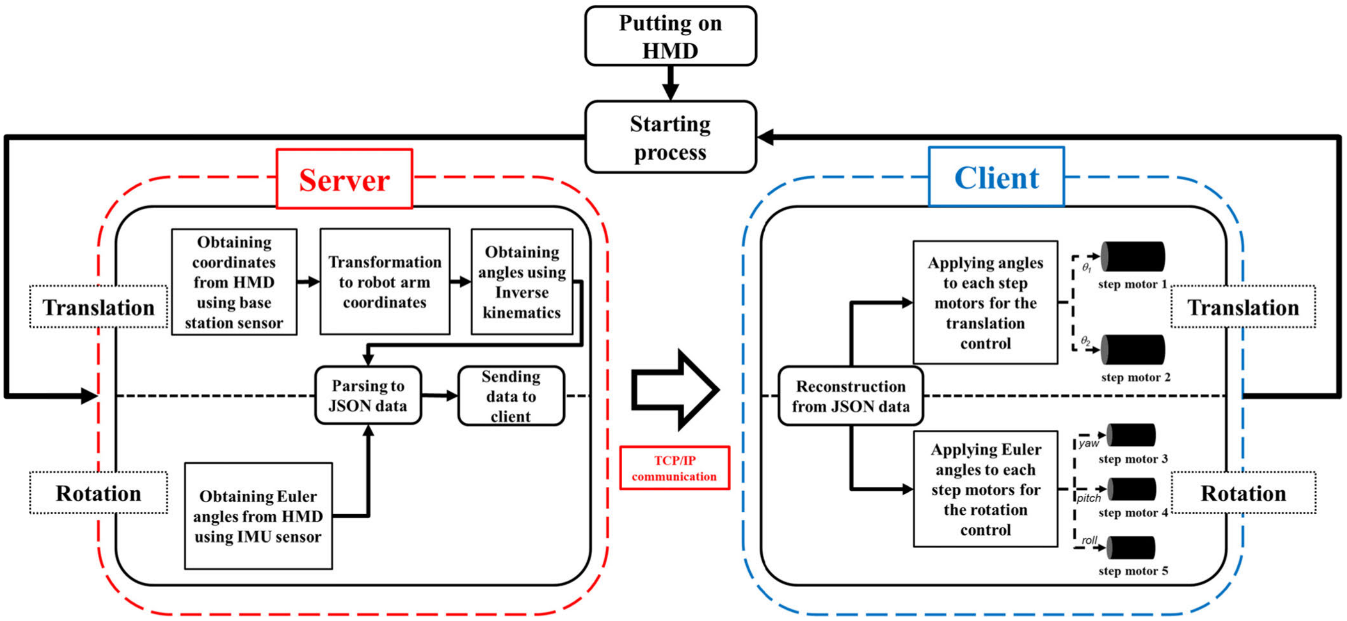
Fig. 4 shows the overall HMD-based robot arm control system. To implement the control of the robot arm system, rotation information and translation information are required. Therefore, in our system, the rotation angles and the translation coordinates of x and y are acquired through the inertial measurement unit (IMU) in the HMD, which is used to track the rotation angle, and the two base stations, which are used to track the translation motion.

The HMD used in the proposed system is the HTC VIVE [11]. Further, three ROBOTIS MX-28 actuators [18] for the camera rotation control and two ROBOTIS L54-30-S500-R actuators [19] for the camera translation control are used in our system. The communication protocol to control the robot arm is the same TCP/IP communication that is used in the real-time image transmission and rendering process, the details of which are described as follows.

1) CAMERA ROTATION CONTROL

Fig. 4 shows the camera rotation control system. As the robot arm control system begins, the rotation data of the HMD are collected on the server PC. Rotation data are based on Euler's principle [20], [23] and have three values: *yaw*, *pitch*, and *roll*. The obtained angle information data of the rotation is sent to the client, and on the client the data received from the server is supplied to the three step motors connected to the PC to synchronize the angular position of the camera with the angular position of the HMD.

This process can be divided into two stages: the angle data acquisition and the motion control of the camera. Initially, the angle data is acquired by updating the HMD direction in real-time using the steam VR plugin SDK [17], [24] of Unity 3D with the HMD connected to the server PC. This data is measured by an IMU sensor inside the HMD. Angle data indicate the rotation angles of *yaw*, *pitch*, and *roll*, respectively. As shown in Fig. 5, if the user rotates his head, the equivalent rotation has to be applied to the camera on the Unity 3D. To facilitate this, the obtained angle data is transmitted to the client. In order to control the direction of the camera module according to the Euler angles as shown in Fig. 5, three step motors are attached to the bottom of the camera. Each motor corresponds to the rotation axis of *pitch*, *roll*, and *yaw* in that order. By applying the direction data to each step motor, the direction of the camera module can be



Putting on HMD
Starting process
Server
Client
Translation
Obtaining coordinates from HMD using base station sensor
Transformation to robot arm coordinates
Obtaining angles using Inverse kinematics
Parsing to JSON data
Sending data to client
Rotation
Obtaining Euler angles from HMD using IMU sensor
TCP/IP communication
Reconstruction from JSON data
Applying angles to each step motors for the translation control
Applying Euler angles to each step motors for the rotation control
θ_1
θ_2
step motor 1
step motor 2
yaw
pitch
roll
step motor 3
step motor 4
step motor 5
Translation
Rotation

FIGURE 4. Block diagram for robot arm control: To control the robot arm, the server processes the coordinates obtained from the movement of the HMD as JSON data and transmits it to the client to control the robot arm thus moving the arm through the distance to same ratio as the HMD's movement.

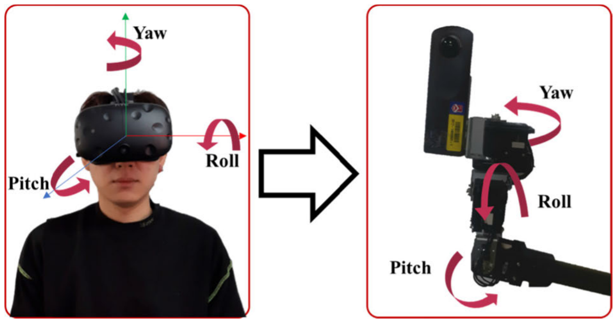


FIGURE 5. Concept for camera rotation angle control system: The Euler angles used to control the camera angles consists of yaw, roll, and pitch, and each of the three step motor applies change values according to the yaw, pitch, and roll.

synchronized with the direction of the HMD. As the direction of the camera is changed, the image acquired in the changed direction is rendered on the HMD. As a result, the user can view the image acquired from the camera module through the HMD in real-time, and can control the direction of the camera by just turning his head without any other special operation.

2) CAMERA TRANSLATION CONTROL

The camera translation control system is shown in Fig. 4. The translation coordinates of the HMD in three-dimensional space can be obtained by using two base station sensors provided by an HTC VIVE VR system and the steam VR plugin SDK of Unity 3D that are the same as those provided for the camera's rotation control.

The acquired translation data is of three types, representing information from each of the x, y, and z axes, respectively. However, because the translation of the robot arm is controlled only by two step motors, it is necessary to transform the three-dimensional coordinates into two dimensional coordinates. To accomplish this transformation, we define the following equation that is used to calculate the intended displacement of the user's HMD.

$$x_{2-d} = \sqrt{x_{3-d}^2 + z_{3-d}^2}, \tag{1}$$

where x_{3-d} and z_{3-d} denote the x and z axes coordinates in the three-dimensional coordinate system, respectively, and x_{2-d} denotes the x axis coordinate in the two-dimensional coordinate system obtained by transformation from the three-dimensional coordinate system. In the two-dimensional coordinates system, the coordinate values of x_{2-d} are defined using the Euclidean distance of the coordinate values of x_{3-d} and z_{3-d}; the y axis value of the two-dimensional coordinate is the same as the three-dimensional y axis value. We designate the x_{2-d} axis as the x axis of the robot arm, and it describes the width of the robot arm as shown in Fig. 6. In addition, we let the y axis on the robot arm represent the height. In order to control the translation of the robot arm, when the motion of the user's head occurs in HMD coordinates, the changed angles should be applied to the two joints of the robot arm constituted by the two step motors. Two conditions are necessary to perform the above procedure. To begin with, users have each individual height. Therefore, if the robot arm does not have a height that fits perfectly the length of the arm, the robot arm may not use the full range of motion. For example, let's assume there are the robot arm with a height of 20 and the user with a height of 20. if the user moves 10, the robot arm is moved by 10. therefore, both of remaining movable ranges of the robot and the user are 10. However, if the user with a height of 15 moves 10,

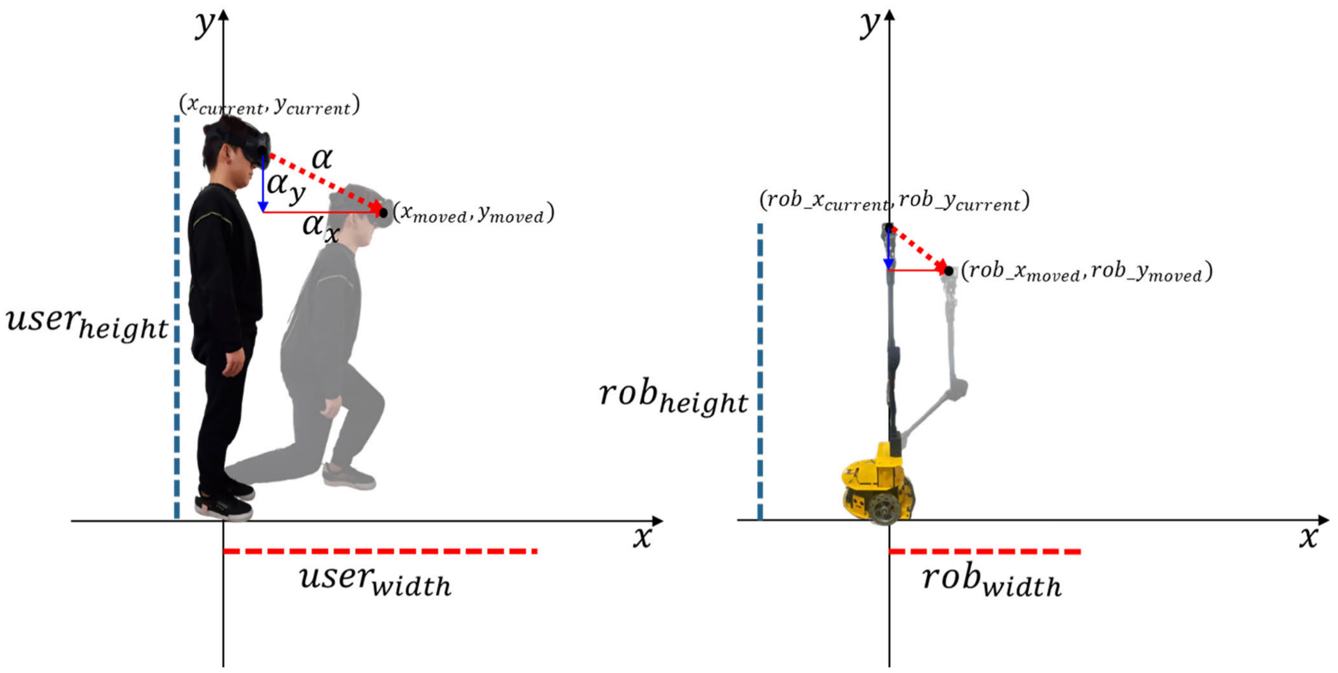


FIGURE 6. Concept for camera translation control system: To control the translation of the camera, the translation is controlled by applying the ratio of the distance the HMD moves to the distance moved by the robot arm to the coordinates of the robot arm.

the remaining range of the robot arm also moves 10. In this case, the remaining movable range of the user is 5 and the robot arm's movable range is 10. therefore, the user can't use full of the height of the robot arm, even if the user moves his whole remaining movable range. Therefore, the height of the user and the length of the robot arm should be proportionally mapped onto each other at the beginning of the robot arm control process. By mapping the height of the user and the height of the robot arm, the translation of the camera can be controlled using the entire length of the robot arm according to the height of the user, which helps in realistic monitoring. Second, after applying the coordinates obtained from the HMD to the coordinates of the robot arm, the angle values to be applied to each step motor should be acquired to control the translation of the robot arm. This is because the translation data is not the value of the angle of movement, but the linear coordinate value. The process is as follows.

To map the height of the user to the length of the robot arm, it is necessary to know the height range of the user (his maximum and minimum heights). Therefore, at the start of this process, the user pulls the trigger of the controller at his maximum height to receive the corresponding maximum coordinate value, and then releases the trigger at his lowest height to receive the corresponding minimum coordinate value. The difference between the highest and lowest heights thus obtained, can be proportionally mapped to the length of the robot arm. After obtaining this range, the user's height and movable range are decided as shown in Fig. 5. Thus, when the user's motion reflects in the coordinates of the HMD, the ratio of the distance moved by the user to the movement in each axis of the two-dimensional coordinates x and y of the robot arm is calculated. Because user's physical characteristics, such as the height of the user, are different for each other, this ratio is used to control the extent of the motion of the robot arm, instead of moving the robot arm through the absolute moving distance value obtained from the HMD. The process can be expressed by the following equations.

$$\alpha_x = x_{moved} - x_{current}, \tag{2}$$

$$\alpha_y = y_{moved} - y_{current}, \tag{3}$$

$$\Delta\alpha_x = \frac{\alpha_x}{user_{width}}, \tag{4}$$

$$\Delta\alpha_y = \frac{\alpha_y}{user_{height}}, \tag{5}$$

where x_{moved} and y_{moved} denote the coordinates x and y of the HMD after the movement, and $x_{current}$ and $y_{current}$ denote the current coordinates, before the movement of the HMD occurs, along the x-axis and y-axis of the HMD, respectively. Therefore, (2) and (3) calculate the distance moved on each axis x and y by the HMD. α_x and α_y represent the coordinates after the movement on each axis, where $user_{width}$ and $user_{height}$ denote the range of the width and height that the user can traverse. (4) and (5) represent the ratio of movement to the possible range of movement for each axis. $\Delta\alpha_x$ and $\Delta\alpha_y$ represent the ratio of moving distance of the HMD for each axis obtained by dividing the distance moved obtained from (2) and (3) by the whole range of movement for each axis. The ratios of the acquired moving distances are applied to the robot arm to facilitate the translation of the robot arm. The motion of the robot arm is governed by the following equations.

$$rob\_x_{moved} = rob\_x_{current} + (\Delta\alpha_x \times rob_{width}), \tag{6}$$

$$rob\_y_{moved} = rob\_y_{current} + (\Delta\alpha_y \times rob_{height}). \tag{7}$$

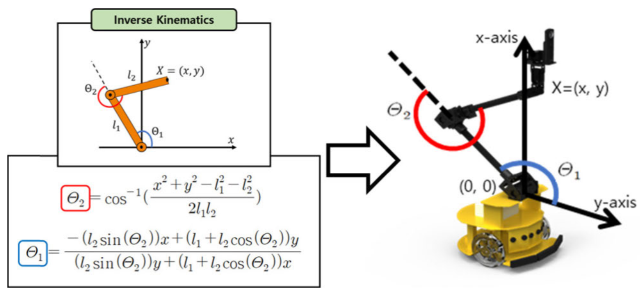

Inverse Kinematics
y
x
l_2
l_1
$X = (x, y)$
Θ_2
Θ_1
$\Theta_2 = \cos^{-1}(\frac{x^2+y^2-l_1^2-l_2^2}{2l_1l_2})$
$\Theta_1 = \frac{-(l_2\sin(\Theta_2))x+(l_1+l_2\cos(\Theta_2))y}{(l_2\sin(\Theta_2))y+(l_1+l_2\cos(\Theta_2))x}$
x-axis
X=(x, y)
(0, 0)
y-axis

FIGURE 7. **Operating principle for inverse kinematics: Principle of applying inverse kinematics to a step motor configured in a robot arm.**

In the equations above $rob\_x_{current}$ and $rob\_y_{current}$ denote the current coordinates, before the motion in the x and y directions of the robot arm. (6) and (7) show the process of applying the proportionally adjusted magnitude of the motion to the coordinates of the robot arm. rob_x_{moved} and rob_y_{moved} represent the coordinate values for the respective axes of the robot arm to which the motion is applied. The movements of the robot arm are calculated by multiplying the motion ratios $\Delta\alpha_x$ and $\Delta\alpha_y$ by rob_{width} and rob_{height}, respectively. Finally, the coordinate values after moving the robot arm are acquired by adding the current coordinates $rob\_x_{current}$ and $rob\_y_{current}$ to the distance over which the robot arm moved, $(\Delta\alpha_x \times rob_{width})$ and $(\Delta\alpha_y \times rob_{height})$, respectively.

In order to perform the motion according to the coordinates of the robot arm, the angles of the two joints constituted by the two step motors should be obtained. In this system, the angles according to the translated coordinate values are obtained by using the principle of inverse kinematics [21]. The principle of operation through inverse kinematic is shown in Fig. 7. l_1 and l_2 denote the length of each linker, respectively. In order to move to the target coordinate represented by x and y, the angle values θ_2 and θ_1 to be applied to each step motor are calculated in order, beginning from the position at the end of the arm. The end of the arm is the point of attachment of the camera. Therefore, it is possible to transmit the obtained angle data θ_1 and θ_2 to the client and control the translation, while the client supplies the angles by which to move to each step motor.

3) DATA FOR ROBOT ARM COMMUNICATION

Unlike the real-time image transmission and rendering system, the robot arm control data communication has the client as a data reception unit and the server as a data transmission unit. All the Euler angle data with *yaw*, *pitch,* and *roll*, and the angle data with θ_1 and θ_2 for translation control need to be preprocessed to transmit five pieces of data at a time. The preprocessing procedure converts the data for the five angles with *yaw*, *pitch*, *roll*, θ_1, and θ_2 into JavaScript object notation (JSON) data so that it can be treated as a single data string. The converted JSON data is sent to the client when both the server and the client are ready. This data communication process is performed by a separate communication with the real-time image system. Because there is a large difference between the frame size of the image and the size of a single piece of JSON data, there is a large difference in speed between the JSON data communication and a single frame data communication. If the JSON data is transmitted with the frame data, the position controlling system of the camera experiences unnecessary deterioration in terms of performance. Therefore, in this system, an additional communication system for the JSON data is established separately from the real-time image transmission system. This new communication system operates independently of the real-time image communication system.

The received JSON data is restored by extracting the angle values at the client to obtain the *yaw*, *pitch*, *roll*, θ_1, and θ_2 values. As shown in Fig. 4, the Euler angle values for rotation control and the angle values for translation control are applied to each step motor attached to the camera, and the above process is repeated.

III. EXPERIMENTAL RESULTS

The proposed system does not allow cut-off renderings or a large latency from the motion controls of the HMD-based robot arm system for the real-time operation. In addition, the robot arm should move the same distance as the user's movement to get images from the desired position. Therefore, two experiments are conducted for the evaluation. Initially, the image transmission time delay was checked to verify the real-time HMD rendering and real-time robot arm control. Next, the distance error was measured to check if the user's movement data acquired from the HMD are precisely applied to the robot arm. These experiments were carried out with the client and server. In the case of the client, the robot arm, which has five step motors, three of them being ROBOTIS MX-28, and two being ROBOTIS L54-30-S500-Rs, held a RICOH Theta S camera on the arm, and a mini PC, namely the intel NUC7i7BNH on the body. The server consisted of an HTC VIVE VR system connected to a PC.

A. DATA TRANSMISSION DELAY MEASUREMENT

All processes in the remote monitoring system should be completed in real-time. If the transmission speed is not above a certain level, users will see choppy and discontinuous videos that will diminish their viewing experience. Therefore, the following experiment measured two transmission delays, the delay in image transmission from the camera to the HMD, and the delay in the data transmission from the HMD to the robot arm, during the control of the robot arm. Decreasing the image resolution is expected to sustain a certain frame rate that can avoid the phenomenon of choppily rendered images. The results are shown in Table 1 and Fig. 8. In this experiment, the average data sizes of various image resolutions, and the average transmission delay of different robotic-arm control data, are measured to compare the relationship between image resolutions and frame rates. Images are groups of three sizes: the size with the original image resolution, the group of sizes with the original resolution of RoI and four times degraded resolution of the original resolution of whole image,

TABLE 1. Transmission delays for different data sequences.

Number of image frame data	Data		Data size (bytes)	Average time step (ms)	Frame per second (fps)
200 frames	Image resolution	1920×960(original)	98745	98.35	10.167
		640×640(RoI) 480×240(whole)	38915	69.73	14.341
		640×640(RoI) 240×120(whole)	32462	64.84	15.423
	Robot arm control data		92	1.07	930.448
400 frames	Image resolution	1920×960(original)	103811	102.12	9.792
		640×640(RoI) 480×240(whole)	39013	70.36	14.213
		640×640(RoI) 240×120(whole)	31414	64.53	15.497
	Robot arm control data		95	1.01	986.777
600 frames	Image resolution	1920×960(original)	99983	99.78	10.022
		640×640(RoI) 480×240(whole)	39236	70.76	14.132
		640×640(RoI) 240×120(whole)	32034	65.12	15.356
	Robot arm control data		93	1.04	958.037
800 frames	Image resolution	1920×960(original)	98804	97.56	10.25
		640×640(RoI) 480×240(whole)	39797	71.21	14.043
		640×640(RoI) 240×120(whole)	31870	64.97	15.392
	Robot arm control data		91	1.01	987.264
1000 frames	Image resolution	1920×960(original)	104325	102.74	9.733
		640×640(RoI) 480×240(whole)	39685	70.89	14.106
		640×640(RoI) 240×120(whole)	32036	65.03	15.378
	Robot arm control data		96	1.06	914.494

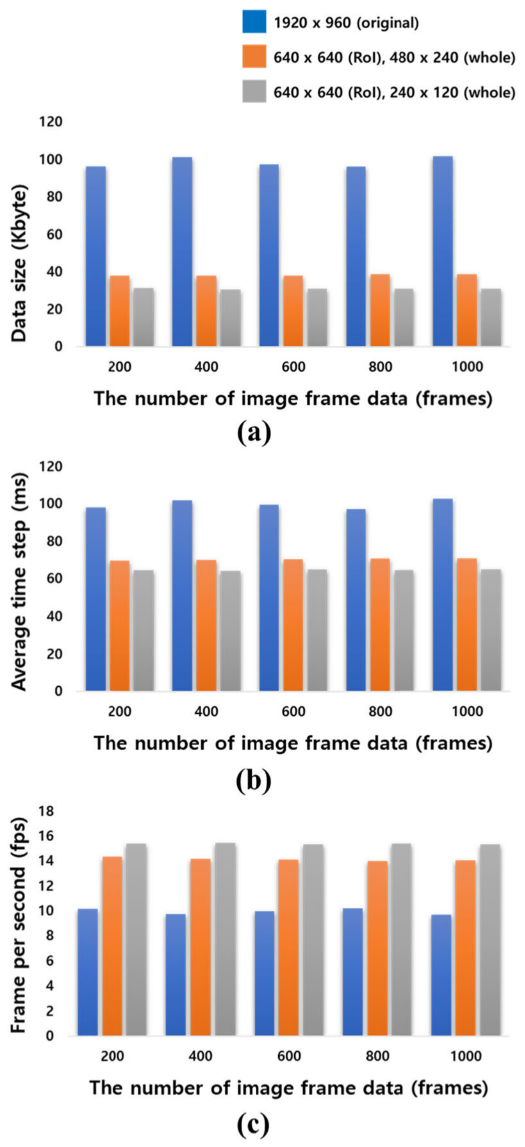


FIGURE 8. Graphs of data transmission delay measurement: (a) the data size, (b) the average time step, and (c) the frame per second.

and the group of sizes with the original resolution of RoI and eight times degraded resolution of the original resolution of whole image. The robot arm control data that are in the JSON file format had five pieces of information that are necessary to control the robot arm. The averages of the data sizes and the transmission delay are obtained through the transmission of more than two hundred packets of data. Based on intuition, we expressed the number of frames per second (fps) at each resolution for checking transmission delay. As the transmission delay increases, the unit of fps is inversely related. In the real-time image transmission issue, the unit of fps is important, so it is preferable to check the transmission delay through fps. In addition, to identify the variation according to the number of pieces of data, we experimented by dividing the data from two hundred data packets into a thousand packets of data.

The image transmission system was able to achieve 9.992 fps on average, over each set of data when transmitting the original resolution image. This implies that the transmission of the original images causes a loss in frame sequencing while being rendered on the HMD, because when we monitored the experiment, we detected that the video looks broken at less than 15 fps. Therefore, it was experimentally confirmed that at least 15 fps is required for a real-time monitoring environment. Therefore, to satisfy the constraint of a minimum frame rate of 15 fps, the system should transmit the original image resolution of RoI and the lower resolution of whole images more than eight times to enable a smooth and continuous rendering of the captured images in the HMD. During the robotic-arm control data transmission, the average transmission delay for each piece of data measured 1.038 ms, leading us to conclude that

TABLE 2. Robot arm translation error for different distances.

distance (cm)	10	30	50	70
HMD (user)	10	30	50	70
Robot arm	9.762	32.28	53.62	63.99
Error rate	2.38%	7.6%	7.24%	8.59%

the control of the robot arm system can be implemented in real-time.

B. DISTANCE ERROR MEASUREMENT

In the proposed monitoring system, the rotation angle information and translation information of the user obtained through the sensors of the HMD are transmitted to the client, after which the client applies the received data to each step motor. In this process, there should be extremely small displacement error in controlling the rotation and translation of the robot arm to facilitate precise monitoring. Table 2 shows the result of the experiment regarding the displacement errors between the data received from the users and the actual movement of the robot arm. In the experiments, the results are obtained by actual measurements that are carried out by using the displacement measurement tool like a tape measure and ruler. The displacement error of the robot arm is represented by the proportional mapping between the length of the robot arm and the user's height. On average, 6.45% of error is observed depending on the distance moved, which can be considered as being due to the error in the sensor accuracy of the HMD and in the step motors.

IV. CONCLUSION

In this study, we have proposed a convenient and realistic camera control system with an HMD equipped the robot arm to give users the ability to manipulate the movable camera easily from a distance to permit remote monitoring. In addition, users can immerse themselves in the monitored scenes from different camera angles using the HMD. Further, we designed a system to control the camera position according to the user's motion using the sensors of the HMD. The proposed remote monitoring system improves the lack of the sense of reality due to the lack of stereoscopic effect and immersion of the conventional monitoring methods. Further, in order to provide the user with the image quality of the original image and to ensure the monitoring system in real-time, we used a method that preserves the image resolution of the region that the user views and reduces efficiently the image resolution of only the remaining areas. The proposed system is used in circumstances where human operators cannot keep direct watch, or can be deployed in hazardous environments and in extreme weather conditions. The experimental results prove that the proposed system is suitable for remote monitoring systems because it satisfies the requirement of a low translation error of displacement, and the requirement that the transmission system must maintain a certain frame rate (fps). Further applications include multi camera modules that can be set up robustly in various environments, and systems that can perform advanced functions with an artificial intelligence (AI)-based computer vision system.

REFERENCES

[1] M. Bramberger, A. Doblander, A. Maier, B. Rinner, and H. Schwabach, "Distributed embedded smart cameras for surveillance applications," *Computer*, vol. 39, no. 2, pp. 68–75, Feb. 2006.

[2] R. Collins, A. Lipton, H. Fujiyoshi, and T. Kanade, "Algorithms for cooperative multisensor surveillance," *Proc. IEEE*, vol. 89, no. 10, pp. 1456–1477, Oct. 2001.

[3] S. Harmon, "The ground surveillance robot (GSR): An autonomous vehicle designed to transit unknown terrain," *IEEE J. Robot. Autom.*, vol. RA-3, no. 3, pp. 266–279, Jun. 1987.

[4] J. White, H. Harvey, and K. Farnstrom, "Testing of mobile surveillance robot at a nuclear power plant," in *Proc. IEEE Int. Conf. Robot. Autom.*, Mar. 2005, pp. 714–719.

[5] D. Di Paola, D. Naso, A. Millela, G. Cicirelli, and A. Distante, "Multi-sensor surveillance of indoor environments by an autonomous mobile robot," in *Proc. 15th Int. Conf. Mechatronics Mach. Vis. Pract.*, 2008, pp. 23–28.

[6] J. Liu, M. Wang, and B. Feng, "IBotGuard: An Internet-based intelligent robot security system using invariant face recognition against intruder," *IEEE Trans. Syst., Man, Cybern. C, (Appl. Rev.)*, vol. 35, no. 1, pp. 97–105, Feb. 2005.

[7] M. Bonetto, P. Korshunov, G. Ramponi, and T. Ebrahimi, "Privacy in mini-drone based video surveillance," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, vol. 4, Sep. 2015, pp. 1–6.

[8] G. Ding, Q. Wu, L. Zhang, Y. Lin, T. A. Tsiftsis, and Y.-D. Yao, "An amateur drone surveillance system based on the cognitive Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 29–35, Jan. 2018.

[9] X. Yue, Y. Liu, J. Wang, H. Song, and H. Cao, "Software defined radio and wireless acoustic networking for amateur drone surveillance," *IEEE Commun. Mag.*, vol. 56, no. 4, pp. 90–97, Apr. 2018.

[10] *Facebook Oculus Specifications and Features*. Accessed: Sep. 2019. [Online]. Available: https://www.oculus.com/

[11] *HTC VIVE Specifications and Features*. Accessed: Sep. 2019. [Online]. Available: https://www.vive.com/

[12] *Microsoft Hololens Specifications and Features*. Accessed: Sep. 2019. [Online]. Available: https://www.microsoft.com/en-us/hololens

[13] J. Rolland and H. Hua, "Head-mounted display systems," in *Encyclopedia of Optical Engineering*. Boca Raton, FL, USA: CRC Press, 2005, pp. 1–13.

[14] *RICOH Theta S Specifications and Features*. Accessed: Sep. 2019. [Online]. Available: https://theta360.com/en/about/theta/s.html

[15] G. K. Wallace, "The JPEG still picture compression standard," *Commun. ACM*, vol. 34, no. 4, pp. 30–44, Apr. 1991.

[16] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz, "Improving TCP/IP performance over wireless networks," *Proc. 1st Annu. Int. Conf. Mobile Comput. Netw.*, Nov. 1995, pp. 2–11.

[17] *Steam VR plugin SDK*. Accessed: Sep. 2019. [Online]. Available: https://assetstore.unity.com/packages/tools/integration/steamvr-plugin-32647

[18] *Specifications and Features*. ROBOTIS MX-28. Accessed: Sep. 2019. [Online]. Available: http://emanual.robotis.com/docs/en/dxl/mx/mx-28/

[19] *Specifications and Features*. ROBOTIS L54-30-S500-R. Accessed: Sep. 2019. [Online]. Available: http://emanual.robotis.com/docs/en/dxl/pro/l54-30-s500-r/

[20] J. Diebel, "Representing attitude: Euler angles, unit quaternions, and rotation vectors," *Matrix*, vol. 58, nos. 15–16, pp. 1–35, 2006.

[21] A. Aristidou, J. Lasenby, Y. Chrysanthou, and A. Shamir, "Inverse kinematics techniques in computer graphics: A survey," *Comput. Graph. Forum*, vol. 37, no. 6, pp. 35–58, Sep. 2018.

[22] *intel NUC Kit NUC7i7BNH Specifications and Features*. Accessed: Sep. 2019. [Online]. Available: https://ark.intel.com/content/www/us/en/ark/products/95065/intel-nuc-kit-nuc7i7bnh.html

[23] S.-W. Choi, S. Lee, M.-W. Seo, and S.-J. Kang, "Time sequential motion-to-photon latency measurement system for virtual reality head-mounted displays," *Electronics*, vol. 7, no. 9, p. 171, Sep. 2018.

[24] M.-W. Seo, S.-W. Choi, S.-L. Lee, E.-Y. Oh, J.-S. Baek, and S.-J. Kang, "Photosensor-based latency measurement system for head-mounted displays," *Sensors*, vol. 17, no. 5, p. 1112, May 2017.

YEOHUN YUN received the degree in computer engineering from Sejong University, South Korea, in 2018. He is currently pursuing the M.S. degree in electronic engineering with Sogang University, South Korea. His current research interests include computer vision and virtual reality.

SEUNG JOON LEE received the B.S. degree in electronic engineering from Sogang University, South Korea, in 2018, where he is currently pursuing the M.S. degree in electronic engineering. His current research interests include computer vision and deep learning.

SUK-JU KANG (Member, IEEE) received the B.S. degree in electronic engineering from Sogang University, South Korea, in 2006, and the Ph.D. degree in electrical and computer engineering from the Pohang University of Science and Technology, in 2011. From 2011 to 2012, he was a Senior Researcher with LG Display, where he was a Project Leader for resolution enhancement and multiview 3D system projects. From 2012 to 2015, he was an Assistant Professor of electrical engineering with the Dong-A University, Busan. He is currently an Associate Professor of electronic engineering with Sogang University. He was a recipient of the IEIE/IEEE Joint Award for Young IT Engineer of the Year, in 2019. His current research interests include image analysis and enhancement, video processing, multimedia signal processing, circuit design for display systems, and deep learning systems.