Rear-Stitched View Panorama: A low-power embedded implementation for smart rear-view mirrors on vehicles

Janice Pan  
The University of Texas at Austin  
janicepan@utexas.edu

Vikram Appia  
Texas Instruments  
vikram.appia@ti.com

Jesse Villarreal  
Texas Instruments  
jesse.villarreal@ti.com

Lucas Weaver  
Texas Instruments  
l-weaver@ti.com

Do-Kyoung Kwon  
Texas Instruments  
d-kwon@ti.com

Abstract

Automobiles are currently equipped with a three-mirror system for rear-view visualization. The two side-view mirrors show close the periphery on the left and right sides of the vehicle, and the center rear-view mirror is typically adjusted to allow the driver to see through the vehicle’s rear windshield. This three-mirror system, however, imposes safety concerns in requiring drivers to shift their attention and gaze to look in each mirror to obtain a full visualization of the rear-view surroundings, which takes attention off the scene in front of the vehicle. We present an alternative to the three-mirror rear-view system, which we call Rear-Stitched View Panorama (RSVP). The proposed system uses four rear-facing cameras, strategically placed to overcome the traditional blind spot problem, and stitches the feeds from each camera together to generate a single panoramic view, which can display the entire rear surroundings. We project individually captured frames onto a single virtual view using precomputed system calibration parameters. Then we determine optimal seam lines, along which the images are fused together to form the single RSVP view presented to the driver. Furthermore, we highlight techniques that enable efficient embedded implementation of the system and showcase a real-time system utilizing under 2W of power, making it suitable for in-cabin deployment in vehicles.

1. Introduction

A National Motor Vehicle Crash Causation Survey (NMVCCS) conducted between 2005 and 2007 showed that the critical reason, or the last failure in the causal chain of events leading up to the crash, of approximately 94% of traffic crashes can be assigned to a driver [1]. Research in the scientific community as well as advancements in the automotive industry have been improving safety systems. Particularly within the past decade, more researchers and manufacturers have been working toward developing suites of intelligent on-board applications, called Advanced Driver Assistance Systems (ADAS), to provide assistance to drivers and ultimately improve safety in vehicles and on the roadways. ADAS applications may serve the purpose of simply showing clearer and more useful displays, or they may even make quick, intelligent decisions for the driver in safety-critical situations.

A driver’s understanding and awareness of their surroundings is integral in keeping them defensive and safe on the road, so an ADAS application that can increase a driver’s field-of-view (FoV) can directly and positively influence the safety of driving. Currently, most cars are equipped with three rear-facing mirrors—two on the vehicle’s exterior by the driver (left) and front passenger (right) side windows and one in the vehicle’s interior, placed front and center and adjusted to capture the scene through the rear window. In order for the driver to gain a full rear-view visualization, they must take their gaze off the road in front and look left, right, and up, which takes critical time and poses a safety hazard.

We present an alternative to the current three-mirror rear-view visualization system that involves a novel camera configuration and a smart panoramic stitching algorithm for creating a single rear-view display, which we refer to as the Rear-Stitched View Panorama (RSVP). The camera set-up (Figure 1) involves single cameras placed on either side of the vehicle, which serve to replace the two external rear-facing mirrors, as well as a stereo pair of cameras placed externally on the vehicle’s rear, which serve to capture the view typically seen through the interior rear-view center mirror. The rear stereoscopic vision provides important information about objects in the rear FoV, which we use to fuse the images as seamlessly as possible.

Another key feature of ADAS systems is reliability. In
particular, the performance of embedded systems, such as in-cabin electronic mirror-replacement systems, can be adversely impacted by heat produced by the embedded processor. Thus, for real-time implementation of our proposed algorithm, we use the TDA3x SoC [2], which is a low-power heterogeneous SoC containing a dual core of ARM Cortex-M4, a dual core of C66x DSP, and a single core of Embedded Vision Engine (EVE) for vector processing. It also provides an integrated hardware Image Signal Processor (ISP) for image signal conditioning and other hardware accelerators, such as a 2-D spatial warp engine, and a hardware Display Subsystem (DSS) for compositing multiple video and graphic output streams. The combination of such a diverse set of compute cores along with the hardware peripherals makes it suitable for our implementation. We showcase the partitioning of the proposed algorithm on the device and highlight the key optimization steps that enabled us to satisfy real-time performance needs.

In this paper, we first discuss previous work related to rear-view visualization (Section 2) in ADAS. Then we give an overview of the proposed system (Section 3), explain how the system is calibrated (Section 4), and describe in detail the method for generating the output RSVP image (Section 5). We also discuss the embedded implementation for a real-time ADAS application (Section 6) and conclude with a discussion of limitations, possible modifications, and future work.

2. Related Work

Vision is an integral sensor modality in ADAS and has been used frequently in past research related to driver assistance applications for relaying important information about the environment and the vehicle’s surroundings to the driver. Common vision-based ADAS applications focus on detection [3, 4, 5, 6, 7, 8, 9, 10], recognition [11, 12, 13, 14, 15], and tracking [16, 17, 18] of objects, with stereo vision being a popular tool in vision-based approaches [6, 7, 8, 9, 10, 12, 13, 20, 21].

Work has also been done specifically to replace the rear-facing cameras with a single easily-accessible display [22], and further, to combine the feeds from each camera to show a single representative rear-view panoramic image [23, 24]. However, no method integrates stereo cameras to extract depth information. Additionally, while previous work has also been done in finding optimal seam lines and developing seamless stitching methods for mosaics and panoramas [25, 26, 27, 28, 29, 30], most of these methods rely on prior calibration, cost-minimization, or point correspondences. No previous work combines all three or uses raw 3D depth estimates in a scene, going beyond simply using point or feature correspondences, in an intelligent seamless stitching algorithm. Our proposed system learns global alignment through an initial calibration step and then uses stereo estimates in a scene, going beyond simply using point or feature correspondences, in an intelligent seamless stitching algorithm. Generating a single panoramic image from three views requires calibration of the system, which enables us to compute each camera’s extrinsic parameters, comprising the

3. System Overview

The RSVP display is meant to replace the rear-view mirror system, which is currently an integral feature of most automobiles. Instead of looking in each of the left, right, and center rear-view mirrors, the driver should be able to look at a single display and see at least the entire FoV that the trio of mirrors is able to provide. The proposed RSVP system uses four cameras: one each on either side to replace the side external rear-view mirrors and a stereo pair of cameras on the rear to replace the center interior rear-view mirror. Feeds from both side cameras are fused with the feed from the reference (left) stereo camera to generate a seamless panoramic image of the rear surroundings. The right stereo camera is only used for stereo depth computations and is not used for visualization. Figure 2 shows an example configuration of the system.

In order to create this seamless display, the captured feeds are transformed into the same visualization coordinate frame. In other words, the system is calibrated, and the frames are transformed independently so that each image looks as if it was captured with the same virtual camera, which in the case of RSVP, may be considered to be located above and in front of the vehicle facing rearwards.

After each image is transformed, they must be combined in such a way that minimizes visual distortion effects and discontinuities in object representation. We refer to the boundaries at which adjacent views are combined as seams. We apply a novel seam-finding method that makes use of the available depth information provided by the stereo camera pair on the vehicle’s rear. Figure 1 shows the process to create the RSVP view. Our stitching algorithm also includes considerations for temporal sequences of frames, maintaining smooth seam-changing transitions so the output frame sequence neither looks jittery nor contains sudden large seam shifts. This feature is represented by the feedback loop in Fig. 1. We also create a single blending look-up table (LUT), using the computed optimal seam lines, to specify the weights used to fuse images together (in the ‘stitching’ block in Fig. 1). Each step in the algorithm, from transforming the input frames to designing the blending LUT, was designed for an efficient embedded implementation, as we explain in the following sections.

4. System Calibration

Generating a single panoramic image from three views requires calibration of the system, which enables us to compute each camera’s extrinsic parameters, comprising the
camera’s location and orientation. Using the extrinsic parameters, each view can be transformed, though likely not without distortions, to appear as if it was taken by a virtual camera with arbitrary location and orientation. This virtual camera is what can be considered as the capture device for the generated panoramic image.

A precisely designed set of three calibration charts with known dimensions and relative locations are placed coplanar with the ground plane for system calibration. As shown in Figure 2, the system is aligned with the calibration patterns such that one full square pattern is visible in each independent camera capture. Determining the locations of the eight corner points of each chart in the camera images provides sufficient correspondences to known world coordinates to calibrate the cameras, because every corner in each captured image has an associated 2D coordinate in the image plane and a real-world 3D coordinate. Using these correspondences, we estimate the homography from the camera image plane to world coordinates using a direct linear transformation. Furthermore, projecting the homography matrix on an orthogonal sub-space can provide the extrinsic pose of the camera in the world coordinate system. Since the points on the chart are measured with physical dimensions, the same physical interpretation is transitioned in the pose estimate of the cameras. Optionally, one can also use additional non-linear optimization approaches to improve the estimation. In our particular example we use the Levenberg-Marquardt approach to refine the camera pose estimations [31]. This calibration is performed offline, and no online calibration is required, because the camera positions are fixed.

5. Panorama Generation

After computing the calibration parameters, projections from each image coordinate system to the world coordinate frame can be computed, and vice versa. The captured frames from each camera can then be transformed into the common coordinate frame of a predetermined virtual camera and fused together. In this section, we discuss the image transformation process, how the optimal seam is selected, and how the three views are stitched together. We also present an approach to maintaining smooth transitions across a temporal sequence of captures.

5.1. Image Transformation

To generate a spatially-coherent output RSVP image, each camera’s intrinsic and extrinsic parameters are used to transform the originally captured images to the virtual camera view. This process involves first projecting each image pixel onto a flat surface behind the vehicle and then using the predetermined virtual camera parameters to project it into the virtual camera image. We can think of the virtual world around the vehicle as represented by a flat surface located some distance behind the vehicle, onto which the appropriate FoV from each camera feed is projected. Other world representations, e.g., cylindrical surfaces, could also be used.

The right stereo camera is only used to compute depth, so we first obtain the disparity map corresponding to the reference image and then use the same transformations on the disparity map that were applied to the left stereo image to obtain a virtual camera ‘view’ of the disparity as well. Figure 3 shows an example set of captured images and the pipeline that generates the output RSVP image. After the images are transformed, the FoV of interest from each image, outlined in red, are combined by stitching along seams, the locations of which we determine with the seam detection algorithm detailed in the following section.

Because we project the RSVP image on a virtual flat surface behind the vehicle, a single projective transform can be computed for each of the three visualization views ($L_{\text{mirror}}$, $L_{\text{stereo}}$, and $R_{\text{mirror}}$). Specifically, for each input from $L_{\text{mirror}}$, $L_{\text{stereo}}$, and $R_{\text{mirror}}$, we use four non-collinear points...
in the RSVP image \( \{p_{oi}\} \), compute the world-coordinate projections of those image points using the virtual camera parameters, and then project those world points back into the input image to get points \( \{p_{o1}\} \). We then compute a projective transformation to map \( \{p_{o1}\} \) to \( \{p_{oi}\} \). We thereby obtain a separate projective transformation matrix for each input, which can efficiently transform input frames as they are continuously being captured.

5.2. Seam Detection and Image Stitching

The output RSVP image is a combination of the views from cameras \( L_{\text{mirror}}, L_{\text{stereo}} \), and \( R_{\text{mirror}} \). Camera \( R_{\text{stereo}} \) is used as part of the stereo system with \( L_{\text{stereo}} \), and the depth information extracted from these two views are used to determine the optimal boundaries along which adjacent views are combined, which we refer to as seam lines.

Because the RSVP image is essentially a projection onto a flat surface with a predetermined depth, when seams are selected such that they pass through objects at the surface distance, those objects appear seamlessly stitched in the output image. In other words, in the process of projecting from each captured image to the virtual surface, a world location that lies on the virtual surface and is captured by adjacent cameras will project back to the same world location, and thus look aligned, in the virtual view. Therefore, we use stereo information to locate points in world space that are close to the projection surface, so that a seam line that runs through their corresponding image points can be computed.

Specifically, we compute disparities for each point on the flat surface, \( D_W \), using its known location relative to the stereo cameras. We then compare this flat surface of disparity values with the transformed disparity map, \( D' \), which was computed between \( L_{\text{stereo}} \) and \( R_{\text{stereo}} \) and corresponds to 3D locations of the scene captured in the stereo cameras. To perform this comparison, we simply take the magnitude of their difference:

\[
D_\Delta = |D' - D_W|, \tag{1}
\]

Therefore, for each seam candidate \( s = \{(x_i, y_i)\} \), defined by a set of pixels in \( D_\Delta \), we define the accumulated matching cost of the seam as \( D_\Delta(s) \) and normalize by the number of pixels in \( s \), for which a valid disparity estimate exists, to obtain cost \( c_s \). Specifically, to compute the cost \( c_s \) of any candidate seam \( s \) with discretized points \( \{ (x_1, y_1), ..., (x_{h_{\text{RSVP}}}, y_{h_{\text{RSVP}}}) \} \), we use the following equation:

\[
c_s = \frac{\sum_{i=1}^{h_{\text{RSVP}}} D'(x_i, y_i) > 0 \ D_\Delta(x_i, y_i)}{|D'(s) > 0|}, \tag{2}
\]

where \( |D'(s) > 0| \) is the number of points in \( s \) with valid disparity estimates, and \( h_{\text{RSVP}} \) is the height of the output panorama in pixels in the transformed domain. Each seam line candidate is specified by \( h_{\text{RSVP}} \) points (one per row). Our output display is 1920 \( \times \) 480, so \( h_{\text{RSVP}} = 480 \) in our implementation. The minimum-cost seam, \( s_{\text{min},W} \), that stitches any two views together is the linear path with the lowest cost that lies within the overlapping region between the views:

\[
s_{\text{min},W} = \arg \min_s c_s. \tag{3}
\]

Figure 5 shows an example of the transformed disparity map and the points in 3D it helped identify as being the appropriate points along which to combine adjacent views as seamlessly as possible.
location, so all seam candidates between the right two image feeds were equally poor choices. In such instances, we instead attempt to stitch along the ground plane, thereby finding a seam that avoids objects in the scene. The same method for determining \( s_{\text{min}, W} \) can be applied to find \( s_{\text{min}, G} \), the min-cost seam along the ground. Instead of matching the surface or ground can be used if one method is preferred over the other. Additionally, when computing matching flat surface disparities and matching the ground disparities, with optional algorithmic constraints. A bias towards using the flat surface disparities (Fig. 6a) is arbitrary. In these cases, we match ground-plane disparities (Fig. 6b).

To select the overall min-cost seam, we select the minimum between the seams computed using matching the flat surface disparities and matching the ground disparities, with optimal algorithmic constraints. A bias towards using the surface or ground can be used if one method is preferred over the other. Additionally, when computing matching flat surface disparities, we can impose a constraint to only allow surface pixels with disparity estimates less than some \( \Delta_w \) to contribute to computing the cost of the seam. Without constraining the valid disparity pixels using \( \Delta_w \), seam selection will be biased towards matching along the ground, because the ground is a dominant feature in the majority of driving scenes, so ground-plane matches are generally more prevalent. This constraint can be used to modify the cost computation of a seam candidate \( s \):

\[
c_s = \sum_{i=1}^{\text{h}_{\text{RSVP}}} D'(x_i, y_i) < \Delta_w \frac{D(x_i, y_i)}{|0 < D'(s) < \Delta_w|},
\]

and the overall min-cost seam is simply

\[
s_{\text{min}} = \min \{ s_{\text{min}, W}, s_{\text{min}, G} \},
\]

which is computed independently for each RSVP frame.

5.3. Temporal Consistency

We also present a method for temporally smoothing the seam movement, because there may be large jumps and jitter in the stitching boundary, which could be distracting and disorienting to the driver, if new seams are computed independently in each frame. To maintain temporal consistency in an RSVP video sequence, we add a feedback loop to our seam-detection algorithm to consider the seam location in the previous frame, as shown in Fig. 1.

With the goal to minimize both unnecessary jitter as well as large jumps in seam locations between frames, we define minimum and maximum distance thresholds, \( p_{\text{min}} \) and \( p_{\text{max}} \), for seam movement. If no points in \( s_{\text{min}} \) are at least \( p_{\text{min}} \) pixels away from the seam in the previous frame \( s_{t-1} \), we do not update the seam, i.e., \( s_t = s_{t-1} \).

However, if any points in \( s_{\text{min}} \) are more than \( p_{\text{max}} \) pixels away from \( s_{t-1} \), then \( s_{\text{min}} \) is scaled such that the largest pointwise displacement between \( s_{\text{min}} \) and \( s_{t-1} \) is \( p_{\text{max}} \). Specifically, if the min-cost seam is \( s_{\text{min}} \) at a time \( t \), and we consider the points comprising each seam:

\[
s_{\text{min}} = \{(x_{1, \text{min}}, y_{1, \text{min}}), \ldots, (x_{\text{h}_{\text{RSVP}}, \text{min}}, y_{\text{h}_{\text{RSVP}}, \text{min}})\};
\]

\[
s_{t-1} = \{(x_{1, t-1}, y_{1, t-1}), \ldots, (x_{\text{h}_{\text{RSVP}}, t-1}, y_{\text{h}_{\text{RSVP}}, t-1})\};
\]

\[
s_t = \{(x_{1, t}, y_{1, t}), \ldots, (x_{\text{h}_{\text{RSVP}}, t}, y_{\text{h}_{\text{RSVP}}, t})\};
\]

(6)

with \( h_{\text{RSVP}} = 480 \), then the \( x \)-coordinates of \( s_t \) can be computed by scaling \( s_{\text{min}} \) as follows:

\[
\Delta x_1 = |x_{1, \text{min}} - x_{1, t-1}|;
\]

\[
\Delta x_{\text{RSVP}} = |x_{\text{h}_{\text{RSVP}}, \text{min}} - x_{\text{h}_{\text{RSVP}}, t-1}|;
\]

\[
r = \max(\Delta x_1, \Delta x_{\text{RSVP}}); \quad p_{\text{max}}
\]

\[
x_{i, t} = x_i, t-1 + \frac{x_{i, \text{min}} - x_{i, t-1}}{r} \quad \forall \ i = [1, 480].
\]

Figure 7 shows examples of how these displacement thresholds can affect the way the new seam is chosen. When
We predefine the blending width, but it could be changed dynamically over frames with knowledge about the scene. Crease from 1 to 0 over a blending width around the seam.

5.4. Stitching

We only require a single blending LUT the same size as the output RSVP image, because the left seam (between $R_{\text{mirror}}$ and $L_{\text{stereo}}$) always falls in the left half of the RSVP image, and the right seam (between $L_{\text{stereo}}$ and $L_{\text{mirror}}$) always falls in the right half of the RSVP image. Thus, we can create the blending LUT under these assumptions and define the weights with respect to the left image of each pair that are being combined. An example of a blending LUT is shown in Figure 8 and red lines are overlayed to indicate seam locations. If the LUT is represented by

$$\{W_{ij}, i \in \{1, \ldots, h_{\text{RSVP}}\}, j \in \{1, \ldots, w_{\text{RSVP}}\}\},$$
then the stitching block (in Fig. 1) will perform the following computations for the left and right halves of the output RSVP image, respectively:

- For $j = \{1, \ldots, \frac{h_{\text{RSVP}}}{2}\}$ and $\forall i$,
  $$\text{RSVP}(i, j) = W_{ij}R_{\text{mirror}}(i, j) + (1 - W_{ij})L_{\text{stereo}}'(i, j),$$
- For $j = \{\frac{h_{\text{RSVP}}}{2} + 1, \ldots, w_{\text{RSVP}}\}$ and $\forall i$,
  $$\text{RSVP}(i, j) = W_{ij}L_{\text{stereo}}'(i, j) + (1 - W_{ij})L_{\text{mirror}}'(i, j).$$

6. Embedded Implementation

Figure 7 shows the data flow for our embedded implementation on the TDA3x SoC. We used four raw Bayer image sensors with a capture resolution of $1280 \times 720$ at 30 fps while the display resolution is $1920 \times 480$ at 30 fps. The image sensors are connected via FPD-Link to the SoC [2], where TDA3x’s hardware ISP processes the video frames, enabling centralized control of exposure and white balance for all camera images. This produces a seamlessly stitched image output with no color or intensity irregularities at the seam (Figures [10] and [11]). These images are then passed to the 2D warp engine for Image Transformation (Section 5.1) and Image Rectification needed for the Stereo Engine [32]. The 2D warp engine is a flexible and efficient back-mapping hardware accelerator for 2D images, which we use for perspective transformations and rectifications of the images and disparity map. By using bilinear interpolation and choosing optimal block sizes, the total compute time for all image
transformations and rectifications on the 2D warp engine was minimized to 29.35 ms, which fits within our 30 fps budget.

The image-rectified stereo pair is passed to the Stereo Engine running on the EVE and DSP1, which creates the disparity map needed for the Seam Detection algorithm in 30.8 ms. We use a stereo baseline of approximately 10 cm and standard semi-global matching for our stereo algorithm, but other dense correspondence methods may be used instead. The Seam Detection and Stitching algorithms require the perspective-transformed stereo disparity map in addition to the transformed images as inputs. In order to meet the real-time latency requirements, we use the disparity map from the previous frame \((t - 1)\) to detect the optimal seam location in the current output frame \(t\).

Furthermore, as described in Section 5.2, we search for the optimal seam by computing a cost on the disparity map for each candidate seam. Here we use a projection surface depth of 15 meters, measured from the stereo camera baseline, i.e., the rear of the vehicle. To reduce the compute complexity we use a coarse-to-fine search strategy, which identifies the optimal vertical seam location and searches for other linear (non-vertical) seams in the small neighborhood of the identified best vertical seam location. Although this strategy reduces computations, a vertical seam search is not cache friendly for an embedded system. To improve cache efficiency, we transpose the disparity map in the 2D warp engine as part of the image transformation step, which converts the search space to horizontal seams.

Since the location of the stitching seam can change in each new frame, we must dynamically generate a new blending LUT. The blending LUT is only computed for the region of the output where the center image overlaps with the left and right images. These optimizations, along with targeted DSP intrinsic optimizations in the blending LUT generation loop, reduce the compute time for the Seam Detection algorithm to 9.5 ms on DSP2.

In the Stitching algorithm, the nonoverlapping portions of the left and right images are simply copied to the output image using efficient hardware Direct Memory Access (DMA) memory transfers, since there is no blending in these regions. For the blended regions, the algorithm was optimized using DSP intrinsic functions to take full advantage of the C66x SIMD architecture. The core software loops performing the alpha-blend operation blends almost two output pixels per one DSP cycle. Since the compute in this case is faster than the memory access rate through the cache, the DMA engine was employed using a ping-pong buffer scheme to bring blocks of both images and the blending LUT from DDR into the DSP L2 RAM and to write the output back from DSP L2 RAM to DDR. With this scheme, the DDR data access time and the blending compute time overlap, thereby reducing the overall time for the stitching algorithm to process each output frame to 4.3 ms on DSP2.

Finally, we also note that the resulting RSVP image that is displayed to the driver should be overlayed with a visualization of part of the vehicle to orient the driver with a sense of relative distances between objects in the scene, as demonstrated in Figure 10. This overlay is fixed and loaded in DDR memory in RGB565 format. The Display Subsystem (DSS) supports reading three video streams and one graphics overlay for color conversion, blending, scaling, and compositing on the final video output port. In our implementation, we use one video stream for the stitched video output and one graphics overlay for the vehicle visualization. The DSS graphics overlay is configured with an alpha blend in hardware which provides an efficient implementation of transparent overlays. With these optimizations, we achieve real-time performance with a system latency of 124 ms.

7. Discussion

Because this RSVP display is intended for automotive applications, safety is, of course, critical. The typical way drivers learn to adjust their rear-view mirrors creates huge blind spots just past the FoV of their periphery, which drivers can only see if they look over their shoulders. The camera configuration in RSVP reduces the extent of this blind spot significantly; however, it also introduces two additional blind spots at either corner of the rear bumper. Figure 12 illustrates this issue. An object right behind the bumper but outside of the FoV of either stereo camera will not be captured by any camera. Additionally, if an object is right next to the bumper and captured in either \(L_{\text{mirror}}\) or \(R_{\text{mirror}}\), stitching the RSVP image together could remove the object from the display entirely, or the object could appear ghosted, both of which are problems. However, if the vehicle is traveling at low enough speeds and/or in reverse, other integral parts of ADAS, such as surround-view, parking assist, or backup assist, would be preferred applications to use over RSVP.

Also, increasingly popular with ADAS is combining information from different sensor modalities. If a radar sensor is able to detect the presence of an object visible in either blind spot captured by \(L_{\text{mirror}}\) or \(R_{\text{mirror}}\) but not captured by the stereo cameras, the seam selection algorithm can be modified to bias the respective seam toward the center of the output RSVP image, which would ensure as much of \(L_{\text{mirror}}\) or \(R_{\text{mirror}}\) is included in the output as possible, so the object will indeed be projected in the RSVP display.

8. Conclusion

We have introduced a novel system involving a new camera configuration and RSVP-generation algorithm to replace the three rear-facing mirrors in most current automobiles. Our camera system comprises (1) two side cameras—one on either side of the vehicle—to replace the side rear-view
mirrors, and (2) a stereo pair of cameras on the vehicle’s rear to replace the center rear-view mirror looking through the rear window and to provide a means of extracting depth information from the scene. After initial calibration of the entire system, transformations of each camera feed can be computed to project the captured images onto a virtual surface, which represents the output RSVP image and can be thought of as being captured by a single rear-facing virtual camera.

The three images used for visualization ($L_{\text{mirror}}$, $R_{\text{mirror}}$, and $L_{\text{stereo}}$) and the disparity map, computed between the stereo image pair, are transformed and passed to our seam-detection algorithm, which uses disparity matching to then determine optimal min-cost seam lines along which to combine adjacent camera frames. Then a single weighting LUT is generated using prior knowledge of where each camera projects in the output, so the images can be combined efficiently. Our embedded implementation of this algorithm demonstrates the efficiency and feasibility of integrating a real-time RSVP display in current ADAS systems.
References


