A Novel Approach for the User Assisted Separation of Reflections From an Image

M.A. Ansari [1] * Member IEEE and R.S. Anand [2]

1. Department of Electrical Engineering, Research Scholar, Indian Institute of Technology Roorkee-247667 India.
2. Department of Electrical Engineering, Associate Professor, Indian Institute of Technology Roorkee-247667 India.

E-mail: m_ahmad786@rediffmail.com, mahmadiitr@gmail.com

Abstract: **In this paper, we present a technique that works on arbitrarily complex images but we simplify the problem by allowing user assistance. We allow the user to manually mark certain edges or areas in the image as belonging to one of the two layers. Separating reflections from a single image is a massively ill-posed problem. We have focused on slightly easier problem in which the user marks a small number of gradients as belonging to one of the layers. This is still an ill-posed problem and we have used a prior derived from the statistics of natural scenes: that derivative filters have sparse distributions. We showed how to efficiently find the most probable decompositions under this prior using linear programming. Our results show the clear advantage of a technique that is based on natural scene statistics rather than simply assuming a Gaussian distribution.**

Keywords: **Linear superposition, Complex Images, Separating reflections, Gradients, Derivative filters, Sparse distributions, ROI, Linear programming.**

I. INTRODUCTION

When we take a picture through transparent glass, the image we obtain is often a linear superposition of two images: the image of the scene beyond the glass plus the image of the scene reflected by the glass.

The Problem: Mathematically, the problem is massively ill-posed. The input image I(x, y) is a linear combination of two unknown images $I_1(x, y)$ and $I_2(x, y)$ given by:

$$I(x, y) = I_1(x, y) + I_2(x, y) \quad \ldots\ldots\ldots\ldots\ldots (1)$$

Obviously, there are an infinite number of solutions to equation (1): the number of unknowns is twice the number of equations. Additional assumptions are needed. On the related problem of separating shading and reflectance, impressive results have been obtained using a single image. These approaches make use of the fact that edges due to shading and edges due to reflectance have different statistics e.g. shading edges tend to be monochromatic. Unfortunately, in the case of reflections, the two layers have the same statistics, so the approaches used for shading and reflectances are not directly applicable [2]. A method was presented that used a prior on images to separate reflections with no user intervention [7]. While impressive results were shown on simple images, the technique used a complicated optimization that often failed to converge on complex images.

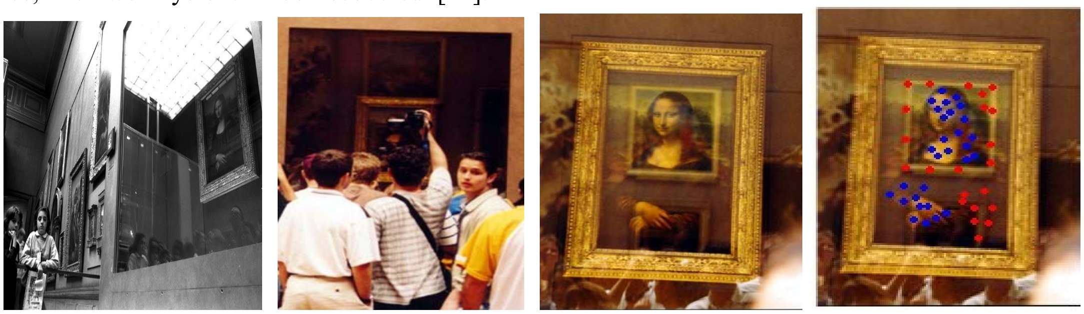
Fig.1(d) shows the Mona Lisa image with manually marked gradients: blue gradients are marked as belonging to the Mona Lisa layer and red are marked as belonging to the reflection layer. The user can either label individual gradients or draw a polygon to indicate that all gradients inside the polygon belong to one of the layers. This kind of user assistance seems quite natural in the application we are considering: imagine a Photoshop plug-in that a tourist can use to post-process the images taken with reflections. As long as the user needs only to mark a small number of edges, this seems a small price to pay.

Even when the user marks a small number of edges, the problem is still ill-posed. Consider an image with a million pixels and assume the user marks a hundred edges. Each marked edge gives an additional constraint for the problem in equation (1). However, with these additional equations, the total number of equations is an only million and a hundred, far less than the two million unknowns. Unless the user marks every single edge in the image, additional prior knowledge is needed.

Following recent studies on the statistics of natural scenes [3], we use a prior on images that is based on the sparsity of derivative filters. We first approximate this prior with a Laplacian prior and this approximation enables us to find the most likely decomposition using convex optimization. We then use the Laplacian prior solution as an initial guess for a simple, iterative optimization of the sparsity prior [9]. We show that by using a prior derived from the statistics of natural scenes, one can obtain excellent separations using a small number of labeled gradients.

The Need: Fig.1(a) shows the room in which Leonardo's Mona Lisa is displayed at the Louvre. In order to protect the painting, the museum displays it behind a transparent glass. While this enables viewing of the painting, it poses a problem for the many tourists who want to photograph the painting fig. 1(b). Fig.1(c) shows a typical picture taken by a tourist[1]: the wall across from the painting is reflected by the glass and the picture captures this reflection superimposed on the Mona-Lisa image.

A similar problem occurs in various similar settings: photographing window dressings, jewels and archaeological items protected by glass. Professional photographers attempt to solve this problem by using a polarizing lens. By rotating the polarizing lens appropriately, one can reduce (but not eliminate) the reflection [1]. As suggested, the separation can be improved by capturing two images with two different rotations of the polarizing lens and taking an optimal linear combination of the two images. An alternative solution is to use multiple input images in which the reflection and the non-reflected images have different motions. By analyzing the movie sequence, the two layers can be recovered [10]. A similar approach is applied to stereo pairs. While the approaches based on polarizing lenses or stereo images may be useful for professional photographers, they seem less appealing for a consumer level application [5]. Viewing the image in figure 1(c), it seems that the information for the separation is present in a single image. Can we use computer vision to separate the reflections from a single image?



(a) (b) (c) (d)

Fig. 1(a)-(b) The scene near the Mona Lisa in the Louvre. The painting is housed behind glass to protect it from the many tourists. (c) A photograph taken by a tourist at the Louvre. The photograph captures the painting as well as the reflection of the wall across the room. (d) The user assisted reflection problem. We assume the user has manually marked gradients as belonging to the painting layer or the reflection layer and wish to recover the two layers.

The Noise Effect: The image at the left of fig.2 (a) has been corrupted by noise during the digitization process. The 'clean' image at the right of fig.2 (a) was obtained by applying a median filter to the image. The image at the top left of fig 2(b) has a corrugated effect due to a fault in the acquisition process. This can be removed by doing a 2-dimensional Fast-Fourier Transform on the image top right of fig2(b), removing the bright spots {bottom left of fig2(b)}, and finally doing an inverse Fast Fourier Transform to return to the original image without the corrugated background {bottom right of fig2(b)}.

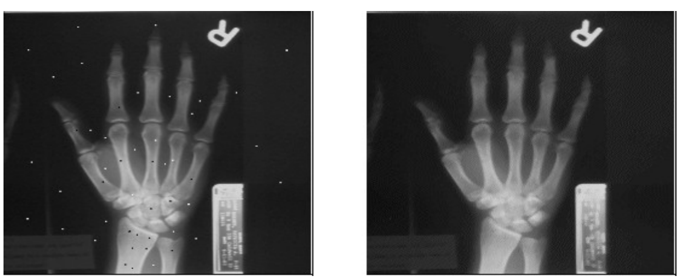


Fig. 2(a) Separation of Noise by median filter

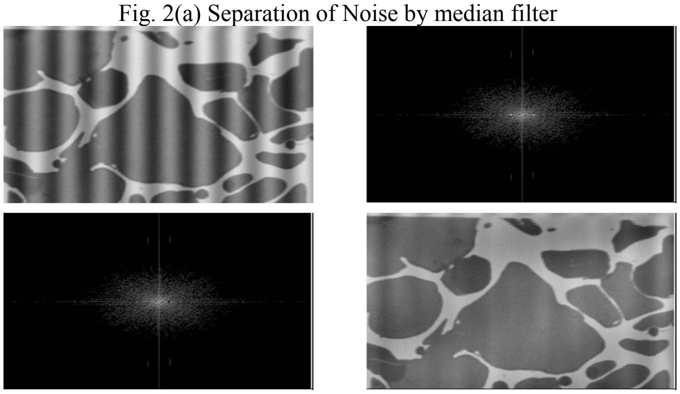


Fig. 2(b) Application of the 2-dimensional FFT

The Solution: A kind of user assistance seems quite natural in the application we are considering: imagine a Photoshop plug in that a tourist can use to post-process the images taken with reflections. Practically speaking, there are more than four Lakh pixels in an average image. And it is not possible for the user to pin point every pixel on the image as belonging to either of the layers. As long as the user needs only to mark a small number of edges, this seems a small price to pay. The probability of a single derivative is given by:

$$\Pr(x) = \frac{\pi_1}{2s_1} e^{-|x|/s_1} + \frac{\pi_2}{2s_2} e^{-|x|/s_2} \qquad (2)$$

Given the histograms over derivative filters, we follow in using it to define a distribution over images by assuming that derivative filters are independent over space and orientation so that our prior over images is given by equation (3):

$$\Pr(I) = \prod_{i,k} \Pr(f_{i,k} \cdot I) \qquad (3)$$

Where f.I denotes the inner product between a linear filter f and an image I, and $f_{i,k}$ is the k^{th} derivative filter centered on pixel i.

The use includes two orientations (horizontal and vertical) and two degrees (i.e. first derivative filters as well as second derivative). The probability of a single derivative is given by equation (2). Equation (3) gives the probability of a single layer. We follow in defining the probability of decomposition I_1, I_2 as the product of the probabilities of each layer (i.e. assuming the two layers are independent).

II. OPTIMIZATION

Now, we are ready to state the problem formally. We are given an input image I and two sets of image locations S_1,S_2 so that gradients in location S_1 belong to layer 1 and gradients in location S_2 belong to layer 2. We wish to find two layers I_1, I_2 such that:

1. The two layers sum to form the input image

$$I = I_1 + I_2$$

2. The gradients of I_1 at all locations in S_1 agree with the gradients of the input image I and similarly the gradients of I_2 at all locations in S_2 agree with the gradients of I.

Subject to these two constraints we wish to maximize the probability of the layers: $Pr(I_1,I_2) = Pr(I_1)\, Pr(I_2)$ given by equation (3). Our approximation proceeds in two steps. We first approximate the sparse distribution with a Laplacian prior. This leads to a convex optimization problem for which the global maximum can be found using linear programming. We then use the solution with a Laplacian prior as an initial condition for a simple, iterative maximization of the sparse prior.

Exactly Maximizing A Laplacian Prior Using Linear Programming: Under the Laplacian approximation, we approximate Pr(I) with an approximate $\tilde{P}r(I)$ defined as:

$$\tilde{Pr}(I) = \prod_{i,k} e^{-|f_{i,k} \cdot I|} \qquad (4)$$

To find the best decomposition under the Laplacian approximation we need to minimize:

$$J(I_1, I_2) = \sum_{i,k} |f_{i,k} \cdot I_1| + |f_{i,k} \cdot I_2| \qquad (5)$$

Subject to the two constraints given above: that $I_1 + I_2 = I$ and that the two layers agree with the labeled gradients. This is an L_1 minimization with linear constraints. We can turn this into an unconstrained minimization by substituting in $I_2 = I - I_1$ so that we wish to find a single layer I_1 that minimizes:

$$\begin{aligned} J_2(I_1) = &\sum_{i,k} |f_{i,k} \cdot I_1| + |f_{i,k} \cdot (I - I_1)| \\ &+\lambda \sum_{i \in S_1, k} |f_{i,k} \cdot I_1 - f_{i,k} \cdot I| \\ &+\lambda \sum_{i \in S_2, k} |f_{i,k} \cdot I_1| \end{aligned} \qquad (6)$$

This minimization can be performed exactly using linear programming. This is due to the fact that the derivatives are linear functions of the unknown image. To see this, define v to be a vectorized version of the image I_1 then we can rewrite J_2 as:

$$J_2(v) = \|Av - b\|_1 \qquad (7)$$

Where $\| \;\; \|_1$ is the L_1 norm, the matrix A has rows that correspond to the derivative filters and the vector 'b' either has input image derivatives or zero so that equation(7) is equivalent to eqn.(6).

Minimization of equation (7) can be done by introducing slack variables and solving:

$$\begin{aligned} &Min : \textstyle\sum_i (z_i^+ + z_i^-) \\ &Subject \;\; to : \\ &Av + (z^+ - z^-) = b \\ &z^+ \geq 0, z^- \geq 0 \end{aligned}$$

The idea is that at the optimal solution one of the variables z_i^+, z_i^- is zero, and the over is equal to $|A_{i\rightarrow} v - b_i|$. The above problem is a standard linear programming one and we use the LOQO linear programming package to solve it.

Optimization Of The Sparse Prior Using Iterated Linear Programming: To find the most likely decomposition under the sparse prior we need to maximize the probability of the two layers as given by equation(3). Using the same algebra as in the previous section this is equivalent to finding a vector 'v' that minimizes:

$$J_3(v) = \sum_i \rho\,(A_{i\rightarrow} v - b_i) \qquad (8)$$

Where ρ(x) is the log probability, ρ(x) is similar to a robust error measure and hence minimizing 'J' is not a convex optimization problem. Nevertheless, using EM we can iteratively solve convex problems. Since we use a mixture model to describe the sparse prior, we can use expectation-maximization (EM) [11] to iteratively improve the probability of decomposition. In the E step we calculate the expectation of hi and in the M step we use this expected value and optimize an expected complete data log likelihood. A standard derivation shows that the EM algorithm reduces to: {E step. calculate two weights w_1, w_2 for every row of the matrix A:

$$w_j(i) \propto \frac{\pi_j}{s_j} e^{-|A_{i\rightarrow} v - b_i|/s_j} \qquad (9)$$

The proportion constant is set so that $w_1(i) + w_2(i) = 1$ for all i, {M step: perform an L_1 minimization given:

$$v^* \leftarrow \arg\min_v \|DAv - Db\|_1 \qquad (10)$$

with D a diagonal matrix whose elements are given by:

$$D(i,i) = w_1(i)/s_1 + w_2(i)/s_2 \qquad (11)$$

At every iteration, we are provably decreasing the cost function J_3 in equation (8). The optimization in the M step was performed using the same linear programming software as in the Laplacian approximation. Three EM iterations are usually sufficient.

III. THE ALGORITHM

1. Load an Image with some regions of Image being fused with reflection.

2. Ask user to mark some points with the help of either point marking or polygon marking. These points are generally marked with a brush thickness ranging from 1 pixel to 10 pixels depending on their relative position in the image and density. The Points should be marked with respect to following scheme:

a. Blue points: User should mark some points on the image as belonging to one of the layers. These points are considered to be coherent part of that image and treated as it while separation the other image from it.

b. Red points: User should mark some other points on the image as belonging to the other layers. These points are considered to be coherent part of the other image and treated as it while separation the other image from the first image.

3. Store these points in an array.

4. Collect the data of image in an array in form of float numbers.

5. Create a derivative filter (-1 0 1) in x direction for the image.

6. The dimension of the filter should be that of image size X image size where image size is the total number of pixels in the image. Store it as Gx.

7. Similarly create 4 more such derivative filters in other direction (y-direction) and other orientations respectively as Gy, Gxx, Gyy and Gxy.

8. Merge all these five derivative filters in one as A1. Remove any filter value at the boundaries of the image as these pixels don't satisfy the balancing of filter as they don't have all the necessary neighborhood pixels.

9. Merge A1 twice to create a matrix (sparse) A (dimensions of A : 10 * image size rows, image size)

10. This A will serve as LHS side of the equation Ax = b;

11. To create the RHS of the equation, apply the above created filter A1 on Image Array (I). Result is an array of size(5 * image size, 1).

12. Now merge the above mentioned array in other array of same size with all entries as zero;

13. Hence now we have RHS of size (10 * image size, 1) and LHS of size (10 image size, image size).

14. Now to bring the effect of marked points into the equation, create an array of 10 image size, and store every element's value as 1.

15. Then multiply rows of A and b with different weights of 4,100 depending on the marked pixels. If the pixel is marked blue than all those rows are multiplied with 100 and those with other color with 4. Now for the remaining 9 *image size rows we multiply each row with some weightage depending on their corresponding pixel in image.

16. Now we are ready with final equation Ax = b. To solve this equation create a numerator by pre multiplying A' with A and pre multiplying A' with b for RHS. This equation is solved by the Matlab equation solver and results are stored in x (image size).

17. This is our Initial guess for the solution. If user himself provides initial guess than no need of this step is there. Directly the values of initial guess are stored in x. The whole result depends on this initial guess.

18. Now we use EM Iteration to refine our solution obtained as initial guess in x;

19. Store the values of A in another sparse matrix oA, b in oB and weightage vector 'f' in oF.

20. Calculate the residual of the solution by putting back the value of x in: power of 'e' to (abs(Ax - b)). If the residual is small for every pixel we move on from our iteration to next otherwise create another vector of size image size E with values of residual calculated in the above step.

21. Solve another equation with LHS as (A' * E * A) and RHS as (A' * E * b).

22. Store these values in the same x.

23. This was M Step of iteration. The above steps 19 to 22 are repeated again one more time.

24. Begin with E step of iteration. Calculate the probability distribution as mentioned in the formulae.

25. Find two weights using above mentioned step 23 for different constant values as of 's' and 'Π' in the formulae.

26. Sum these weights and take weighted average.

27. Store these into a diagonal matrix and pre multiply this matrix with A and b (in the same passion as we multiplied weights for the initial guess). Store the results in oA and ob respectively.

28. Now we repeat the above mentioned steps of EM Iteration 18 to 26 for n number of times where n ranges from 3 to 15 depending upon the quality of the output and time of execution.

29. The values in x are moderated (to remove any -ve entry) and results are displayed in the form an Image.

30. To calculate the other image (reflection) subtracts values of x from I (Problem Image) and display moderated result in another image.

31. Hence we are left with two Images with original and reflection image separated from the problem image.

IV. RESULTS & DISCUSSIONS

We show results of our algorithm on images of scenes with reflections. Four of the images were downloaded from the internet and we had no control over the camera parameters or the compression methods used. For color images we ran the algorithm separately on the R, G and B channels. Fig(3) shows the input images with labeled gradients and our results.

(a) (b) (c) (d)

Figure 3. Results: (a) input image with reflections. (b) Identifying reflections by red and blue points (c) Decomposition and separation of reflections (d) Image without reflections after separation of reflections

The Laplacian prior gives good results although some ghosting effects can still be seen (i.e. there are remainders of layer 2 in the reconstructed layer 1). These ghosting effects are fixed by the sparse prior. Good results can be obtained with a Laplacian prior when more labeled gradients are provided. The non sparse nature of the Gaussian distribution is highly noticeable, causing the decomposition to split edges into two low contrast edges, rather then putting the entire contrast in one of the layers. Fig.(4) shows the software results of the Input image & separation of reflections. In fig. (5), fig.(6) and fig.(7) the algorithmic technique was applied for removing shading artifacts. For this problem, the same algorithm was applied in the log-domain and the results obtained show almost the complete removal of the shading artifacts.

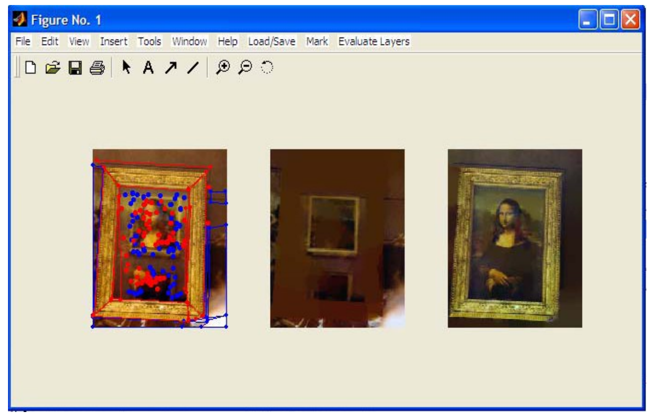


Figure 4. Input image & separation of reflections.

(a) (b) (c) (d)

Figure 5. Removing shading artifacts (a) original image. (b) Labeled image. (c-d) decomposition.

(a) (b) (c)

Figure 6. Results: (a) input image. (b) decomposition (c) Separation of reflection

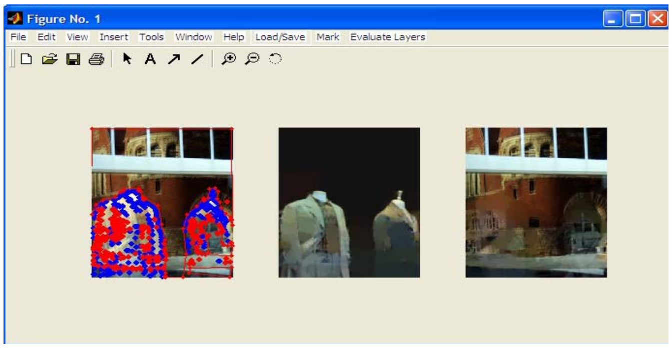


Fig 7. Input image & separation of reflections by pointing method.

Since we are using an off-the-shelf linear programming package, we are not taking advantage of the spatial properties of the optimization problem. The current run time of the linear programming for images of size 240x320 is a few minutes on a standard PC. We have not performed an extensive comparison of linear programming packages so that with other packages the run times maybe significantly faster. We are currently working on deriving specific algorithms for minimizing L_1 cost functions on image derivatives. Since this is a convex problem, local minima are not an issue and so a wide range of iterative algorithms may be used. In preliminary experiments, we have found that a multigrid algorithm can minimize such cost functions significantly faster. We are also investigating using a mixture of Gaussians rather than a mixture of Laplacians to describe sparse distributions. This leads to M steps in which L_2 minimizations need to be performed, and there are a wide range of efficient solvers for such minimizations.

We are also investigating the use of other features other than derivatives to describe the statistics of natural images. Our experience shows that when stronger statistical models are used, we need less labeled points to achieve a good separation. We hope that using more complex statistical models will still enable us to perform optimization efficiently. This may lead to algorithms that separate reflections from a single image, without any user intervention.

Acknowledgements: It is my pleasure to express sincere thanks to the management of College of Engineering Roorkee for providing me leave for pursuing Ph.D at IITRoorkee and Dr. R.S.Anand, Department of Electrical Engineering, IIT Roorkee for providing his constant support and guidance for my research work. I am also thankful to the Department of Electrical Engineering, IIT Roorkee for providing the necessary facilities to carryout this research work.

V. REFERENCES

[1] H. Farid and E.H. Adelson, "Separating reflections from images by use of independent components analysis", Journal of the optical society of America, 16(9):pp.2136-145, 1999.

[2] G. D. Finlayson, S. D. Hordley, and M. S. Drew, ''Removing shadows from images", European Conf. on Computer Vision, 2002.

[3] Levin, A. Zomet, and Y. Weiss, "Learning to perceive transparency from the statistics of natural scenes", Advances in Neural Information Processing Systems, Vol 15, Edn. 2002.

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data ",via the EM algorithm. J. R. Statist. Soc. B, 39:1-38, 97.

[5] Y. Shechner, J. Shamir, and N. Kiryati, "Polarization-based decorrelation of transparent layers: The inclination angle of an invisible surface", Int. Conf. on Computer Vision, pp 814-819, 1999.

[6] E.P. Simoncelli, ''Bayesian denoising of visual images in the wavelet domain", P Mller and B Vidakovic, editors, Wavelet based models, 1999.

[7] R. Szeliksi, S. Avidan, and P. Anandan, "Layer extraction from multiple images containing reflections and transparency", Conf. on Computer Vision and Pattern Recognition, 2000.

[8] M. Tappen, W.T. Freeman, and E.H. Adelson, "Recovering intrinsic images from a single image", In S. Becker, S. Thrun, and K. Obermayer, editors, Advances in Neural Information Processing Systems 15, 2002.

[9] M.J. Wainwright, E.P. Simoncelli, and A.S. Willsky, ''Random cascades of Gaussian scale mixtures for natural images", Int. Conf. on Image Processing, pages 260-263, 2000.

[10] Y. Weiss, ''Deriving intrinsic images from image sequences", Proc. Intl. Conf. Computer Vision, pp.68-75. 2001.

M.A.Ansari: The author is associated with the College of Engineering Roorkee as an assistant professor in the Department of Electrical Engineering and presently pursuing research in the Department of Electrical Engineering, Indian Institute of Technology Roorkee, India. The author is member of IEEE and has a wide teaching and industry experience.

R.S.Anand: The author is working as associate professor in the Department of Electrical Engineering , Indian Institute of Technology Roorkee, India. The author has long experience of medical imaging and Ultrasonics along with Image Processing.