Image Processing for a High-Resolution Optoelectronic Retinal Prosthesis

Alon Asher*, William A. Segal, Stephen A. Baccus, Leonid P. Yaroslavsky, and Daniel V. Palanker

***Abstract*—In an effort to restore visual perception in retinal diseases such as age-related macular degeneration or retinitis pigmentosa, a design was recently presented for a high-resolution optoelectronic retinal prosthesis having thousands of electrodes. This system requires real-time image processing fast enough to convert a video stream of images into electrical stimulus patterns that can be properly interpreted by the brain. Here, we present image-processing and tracking algorithms for a subretinal implant designed to stimulate the second neuron in the visual pathway, bypassing the degenerated first synaptic layer. For this task, we have developed and implemented: 1) A tracking algorithm that determines the implant's position in each frame. 2) Image cropping outside of the implant boundaries. 3) A geometrical transformation that distorts the image appropriate to the geometry of the fovea. 4) Spatio-temporal image filtering to reproduce the visual processing normally occurring in photoceptors and at the photoreceptor-bipolar cell synapse. 5) Conversion of the filtered visual information into a pattern of electrical current. Methods to accelerate real-time transformations include the exploitation of data redundancy in the time domain, and the use of precomputed lookup tables that are adjustable to retinal physiology and allow flexible control of stimulation parameters. A software implementation of these algorithms processes natural visual scenes with sufficient speed for real-time operation. This computationally efficient algorithm resembles, in some aspects, biological strategies of efficient coding in the retina and could provide a refresh rate higher than fifty frames per second on our system.**

***Index Terms*—Biomedical image processing, macular degeneration, retinal prosthesis, retinitis pigmentosa.**

I. Introduction

MANY cases of intractable vision loss arise from selective photoreceptor degeneration. Retinitis pigmentosa (RP), for example, causes the loss of up to 95% of the photoreceptor layer, but spares up to 80% of the inner nuclear layer and $\sim$30% of the ganglion cell layer [1]. Similarly, patients with age-related macular degeneration (AMD) can lose up to 70% of photoreceptors with no loss of other retinal cell types [2], [3]. Approximately 1 in 4000 newborns inherits the genotype for RP, while AMD, which arises from multiple causes including natural aging and environmental stresses, is diagnosed in 700 000 Americans annually. No treatment currently exists for RP, while progression of AMD can be slowed but not prevented.

In cases of selective photoreceptor loss, a potential treatment is to bypass the damaged neural tissue, using electrical stimulation to artificially deliver visual information to the surviving retina. It has been shown in cat models that information can be transmitted from a retinal implant to the brain with temporal resolution corresponding to video rate (40 ms or 25 Hz) and spatial resolution sufficient for simple object recognition [4], [5]. In early human trials, simple visual percepts such as spots and patterns have been produced by electrically stimulating the degenerated retina with just a few electrodes [1], [6], [7]. Recently, we reported a design for a high-resolution retinal prosthetic implant containing up to 18 000 pixels on a 3-mm disk, geometrically corresponding to a visual acuity of up to 20/80 within 10° of visual field [8], [9]. This strategy has an inherent requirement: replacing a damaged part of the nervous system with an electronic system requires that lost neural information processing be replaced as well [10], [11].

A potential complication of this approach is that remodeling of retinal architecture in the absence of photoreceptors signaling can lead to a progressive retinal miswiring [12]. However, highly resolved chronic electrical stimulation may induce neuronal migration and rewiring so as to utilize the provided stimulation [13]. To properly transmit visual signals through the optic nerve, any such remodeling must not drastically change the encoding of visual input in the inner retina.

Here, we describe a set of real-time computer algorithms to restore lost visual processing and prepare images for transmission to the implant placed in the fovea. The fovea is chosen because it supports the highest acuity vision, and because the representation of the fovea is greatest in the visual cortex [14]. This software tracks the location of the implant, crops away unnecessary visual information, distorts the image appropriately for the neural circuitry of the fovea, and applies spatio-temporal filters characteristic of the missing first visual synapse.

In our hardware design (Fig. 1) described previously [8], [9], a video camera transmits 640 $\times$ 480 pixel images at 25 Hz to a pocket PC [8]. The computer processes the data as described in this report, and displays the resulting images on an LCD matrix of similar resolution, mounted on goggles worn by the patient. The LCD screen is illuminated with a pulsed near-infrared (near-IR) light, projecting the images through the eye and onto the retina. The infrared (IR) light is then received by photodiodes on an implanted chip, which is approximately 3 mm in diameter and centered on the fovea. Each photodiode converts

Manuscript received July 4, 2006; revised January 6, 2007. This work was supported in part by the Air Force Office of Scientific Research (MFEL program), and in part by AMO, Inc. and Optobionics, Inc. *Asterisk indicates corresponding author.*

*A. Asher is with the Department of Interdisciplinary Studies, Faculty of Engineering, Tel Aviv University, Tel Aviv, Ramat Aviv 69978, Israel (e-mail: alon.asher@gmail.com).

*W. A. Segal is with the Neuroscience Program, Stanford University, Stanford, CA 94305 USA (e-mail: wsegal@stanford.edu).

S. A. Baccus is with the Department of Neurobiology, Stanford University School of Medicine, Stanford, CA 94305 USA (e-mail: baccus@stanford.edu).

L. P. Yaroslavsky is with the Department of Interdisciplinary Studies, Faculty of Engineering, Tel Aviv University, Tel Aviv, Ramat Aviv 69978, Israel (e-mail: yaro@eng.tau.ac.il).

D. V. Palanker is with the Department of Ophthalmology and Hansen Experimental Physics Laboratory, Stanford University, Stanford, CA 94305-4085 USA (e-mail: palanker@stanford.edu).

Digital Object Identifier 10.1109/TBME.2007.894828

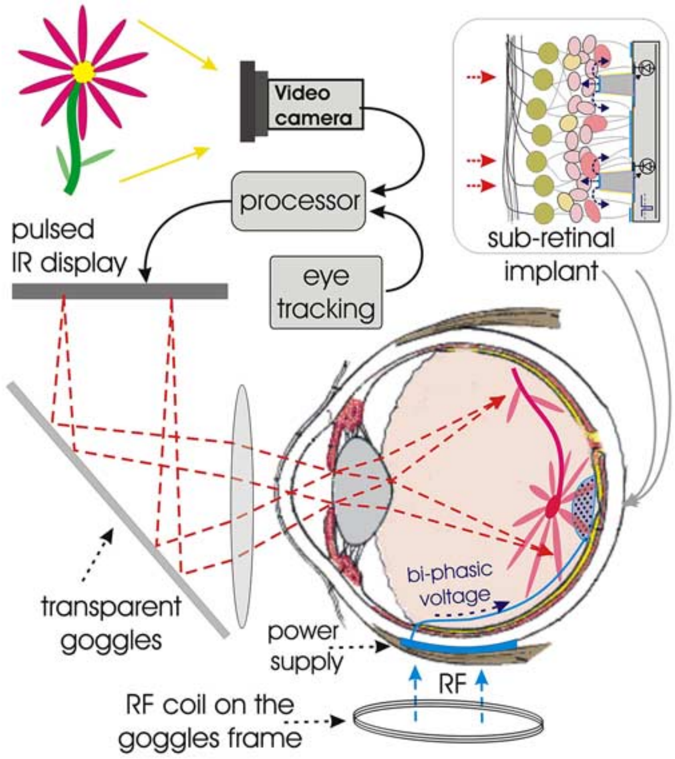


Fig. 1. Prosthetic system design. Images captured by the scene camera are processed and broadcast through a pulsed IR LCD display. The IR light reflects from goggles worn by the patient and is projected through the eye optics onto the chip. Inductive coils provide power to the chip, which lies in the subretinal space and interfaces primarily with inner nuclear layer cells. Each pixel in the chip converts IR light into pulsed biphasic current using a common power line.

the IR signal into a proportional electric current using a common pulsed biphasic power supply. Electrical stimulation by the chip introduces visual information into diseased retinal tissue, while any remaining peripheral vision responds normally to visible light passing through the transparent goggles.

High resolution retinal stimulation can be achieved only if the electrode-to-target-cell distances are small, on the order of the electrode diameter [8], [15]. To achieve a proximity of better than 50 $\mu\mathrm{m}$, we developed 3-D subretinal implants that induce retinal tissue to migrate very close to the stimulation sites [16]. The implant is placed in the subretinal space so as to predominantly stimulate bipolar cells, second-stage retinal neurons that normally receive input from photoreceptors. Although subretinal insertion is more challenging surgically than epiretinal placement, by positioning the implant as early as possible in the visual system, the software need only compensate for the first missing synaptic layer. In addition, more accurate computational models exist of visual processing in the first synaptic layer than exist for the complex neural circuitry of the second synaptic layer in the inner retina.

Because visual information is projected onto the implant through the natural eye optics, the patient can shift gaze naturally about the image. Although the scene video camera captures 30° of the visual field and the IR LCD screen can display the whole image, the chip covers only 10° on the retina. A second, tracking video camera continually monitors direction of gaze or, in other words, the position of the chip in the field of view. Based on the chip's location, the IR LCD screen displays an image spanning only 10°, corresponding to the chip's instantaneous location. Cropping away the parts of the image that would not fall on the implant reduces tissue heating from IR light and saves computational resources.

The input to the signal processing software consists of video images from the scene camera (gray-scale, 8-bit VGA with 640 $\times$ 480 pixels, corresponding to a 30° field) and the tracking camera. The output of the software is an image of 213 $\times$ 213 pixels corresponding to a 10° field, which is transmitted by the IR LCD display (also having VGA resolution). The IR image projected onto the implant is sampled at the resolution of the stimulating array and converted into pulsed electric current in each pixel. The resolution of the retinal implant is lower than that of the LCD screen: each pixel of the screen is projected onto a 14 $\mu\mathrm{m}$ square, while pixels in the retinal implant are at least 20 $\mu\mathrm{m}$ in size. In comparison, foveal cones are normally spaced about 3 $\mu\mathrm{m}$ apart. From each cone, the highest-resolution visual information travels in parallel through a single ON-type and OFF-type "midget" bipolar cell. In addition, information travels through other bipolar cells at lower resolution [17].

Light adaptation, an important function normally performed in part by photoreceptors, is implemented by the automatic gain of the scene camera. For each frame, the signal processing software accomplishes five tasks.

1) Determine the location of the implant from the tracking camera image.
2) Crop away visual data that would not fall upon the chip.
3) Distort the image appropriate to the geometry of the fovea. In the normal foveal center, bipolar cells are pushed aside to reduce loss of acuity from light scattering [Fig. 3(A)]. Thus, to accommodate the foveal anatomy, where there are no bipolar cells, an algorithm performs a geometric "stretching" of part of the image.
4) Implement visual processing of the first synapse. At the level of the photoreceptor to bipolar cell synapse, horizontal cells feed back to photoreceptor synaptic terminals. The three cells are arranged in a synaptic triad that enhances spatial edges and temporal change. This transformation is closely approximated by a linear spatio-temporal filter. Because this synaptic processing is lost when photoreceptors are lost, this spatio-temporal filter is implemented in software.
5) Convert the filtered visual information into an appropriate pattern of electrical current. Although the implant stimulates with short electrical pulses, bipolar cells normally signal with a continuous modulation of membrane potential. Thus, the final step is to choose a pattern that will most closely generate a continuous output from bipolar cells.

A critical requirement of this algorithm is computational efficiency. To provide a naturally appearing sequence of images, the software must process 640 $\times$ 480 pixel images at a frame rate of 25 Hz. To that end, parts of some tasks are carried out in preprocessing algorithms, resulting in fast "lookup tables" referenced during separate real-time algorithms. Previous reports on retinal prostheses have not discussed whether necessary image processing can be carried out in real-time. The present algorithm and software implementation is efficient enough that it is expected to execute on a standard portable microcomputer (pocket PC) in real-time at a video frame rate of 25 Hz.

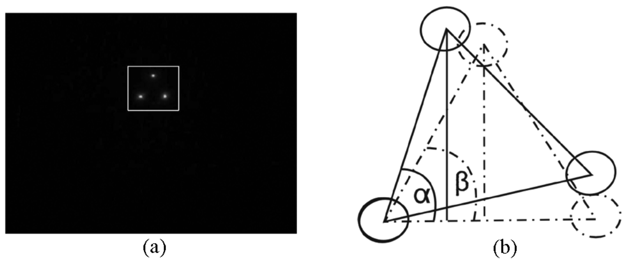


Fig. 2. Tracking algorithm. (A) Image captured by tracking camera. Three reference spots indicate the position and orientation of the chip. The tracking window (white rectangle) contains the reference spots with some additional peripheral space, such that, during fixational eye drift, the reference spots in the next frame will be contained within the same window. (B) Extraction of torsion angle, calculated as $\alpha - \beta$.

II. Software Algorithms

A. Tracking

The tracking algorithm provides the instantaneous position and torsion angle of the chip for later use by the image processing software. As the center of gaze shifts, the chip moves with the retina. As it does, the tracking camera captures light from three reference points on the chip's anterior surface. These reference points might be mirrored pyramidal indentations that reflect some IR light back through the pupil and into the gaze-tracking system. We previously calculated that current video cameras could track the chip's location at 25 Hz and with a precision up to 10 $\mu\mathrm{m}$, about half the width of a pixel on the chip [8].

The input to the tracking algorithm is a stream of video frames, from the tracking camera, containing images of the three reference points [Fig. 2(A)]. The triangle formed by these points can move translationally and torsionally. During gaze fixation, eye position drifts at an average angular velocity of 0.5° per second, up to 2° per second [18]; in the 40 msec between tracking camera frames, the amount of change in the direction of gaze is limited by 0.1°. Therefore, given a 640 × 480 tracking camera image size covering 30° of visual field, the maximal movement of the triangle between sequential frames during eye fixation is less than three pixels. For each frame, we can define a tracking window [Fig. 2(A)] that contains the triangle plus three pixels of vertical and horizontal free space around each vertex of the triangle. The size of the tracking window depends on the area of the triangle and the patient's speed of eye scanning. In our simulation, it was 80 × 88 pixels, approximately $(1/6)^2 = 1/36$ of the movie frame size. The tracking algorithm saves time in the subsequent frame by searching for the shifted reference points only within this window.

Large, gaze-shifting eye movements, known as saccades, can reach hundreds of degrees per second, and will shift the implant out of the reduced search window. When scanning a scene, saccades normally occur at a frequency of less than three times per second. Although saccades are under voluntary control, they occur at a slightly higher frequency in cases of damage to the central visual field [19]. If the three tracking points are not found within the reduced window, the algorithm searches for them within the whole tracking camera image, re-initializing the tracking window to their new location.

Smooth pursuit eye movements used to track a moving object can also reach speeds exceeding tens of degrees per second [20]. However, the opportunity exists to track these smooth eye movements predictively, shifting the tracking window an amount equal to the current measured velocity.

We arbitrarily define the top reference spot's coordinates as the "location" (p, q) of the implant. We calibrate the tracking algorithm by defining $(p_0, q_0) = (0, 0)$ and $\theta_0 = 0$ during the first frame. Fig. 2(B) illustrates the computation of the torsional angle θ as $\alpha - \beta$, where α and β are easily computed using the coordinates of the triangles' vertices.

For each input video frame, therefore, the tracking algorithm performs two tasks. 1) It extracts the positions of the reference points, reporting the coordinates of the top spot as the position (p, q) of the chip and calculating the torsional angle θ as described.

2) It adjusts the tracking window so that the current triangle lies in the center. In the context of gaze fixation, this guarantees that the triangle in the next frame will also remain within the tracking window.

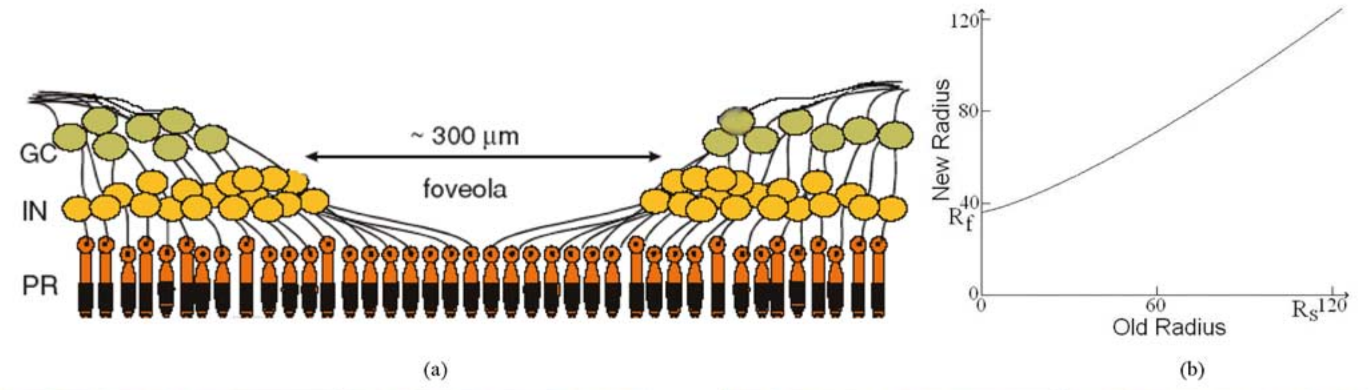
B. Cropping

The scene video camera captures approximately 30° of visual field, but the chip only covers 10°. To increase the efficiency of image processing, the software crops away parts of the image that would not fall on the chip. This also diminishes the amount of light emitted by the IR LCD display, thereby decreasing its power requirements and the heating of intraocular tissue. As the chip moves within the eye, the tracking algorithm reports the position and the rotational angle of the chip. The cropping algorithm identifies a rectilinear window entirely containing the chip and then sets to zero the values of all pixels outside this window. Since the algorithm removes all but 10° of visual field from the 30° captured by the scene camera, subsequent processing algorithms increase in efficiency by operating on one ninth as many pixels.

C. Geometrical Transformation

In the normal retina, photoreceptors lying in the fovea spread radially to contact bipolar cells outside the foveal center, or foveola Fig. 3(A). Since there are no bipolar cells within the foveola for the chip to stimulate, we deliver no IR light to that region. The system must instead reroute the visual information that would fall on the foveola to bipolar cells outside the foveola, just as the photoreceptors reroute that information naturally. As in a healthy retina, the information that would arrive near the foveal center is delivered to bipolar cells close to the fovea's edge, while information that would arrive near the foveal periphery goes to bipolar cells further away.

For the geometrical transformation, we define a "coordinates mapping function" F to approximate the natural spread of visual information. Assuming radial symmetry, the sole argument of F is a point's radial distance from the foveal center. If r is the distance of a photoreceptor from the foveal center, then F(r) is the distance between the foveal center and the bipolar cell that would normally receive a synapse from that photoreceptor. Fig. 3(B) gives an example of a coordinates mapping function. Coordinates in the input frame that are located within the interval $[0, R_s]$ are mapped to the interval $\lfloor R_f, R_s \rfloor$, where R_f



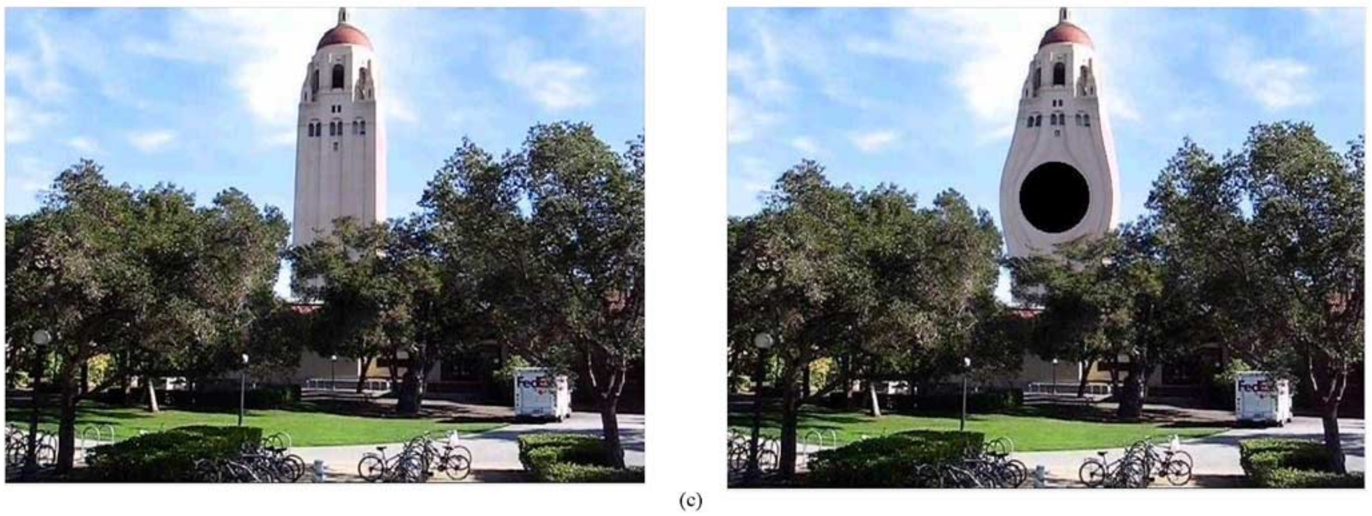


Fig. 3. Geometric transformation. (A) Retinal morphology at the fovea. PR = photoreceptor layer; IN = inner nuclear layer; GC = ganglion cell layer. Photoreceptors in the foveola connect to bipolar cells spread to the side. (B) Coordinates mapping function. Based on the "old radius," the distance of a pixel from the foveal center, a pixel's value is assigned to a larger "new radius" from that center. Thus, image data near the center of the fovea is rerouted farther away to appropriately match the anatomical spreading. Far from the center, at radius R_s, the new radius is equal to the old radius; where the retina does not exhibit any spreading, the mapping function does not reroute any information. (C) Geometric transformation applied to an image. The part of the image that would fall upon the foveola is black because there are no bipolar cells for the chip to stimulate there. The visual information in the foveal area appears stretched, radially symmetrically, around a black spot of radius R_f

is the radius of the fovea and R_s is some radial distance substantially outside the fovea. In practice, pixels in the input frame within R_f of the origin are visually distorted in the output frame around a black circle of radius R_f [Fig. 3(C)].

When image geometrical transformations are required, a common method is to map output image samples onto the input image coordinate system. That is, instead of considering each input pixel and finding the output pixel to which its value should be routed, we consider each output pixel and calculate which input-frame point will dictate its value. This technique is termed "backward coordinate mapping" [21]. The mapped coordinates in the input image coordinates system, calculated with backwards coordinates mapping, may not have integer coordinates. However, the actual input images from the scene camera consist solely of pixels, which, by definition, do have integer coordinates. To relate these actual input pixel values to the values of the noninteger-coordinate input points we need, we use interpolation. For an arbitrary pixel (j, k), let I(j, k) be the intensity of that pixel—the pixel's value. Using bilinear interpolation in view of its low computational complexity, we express the value $I(\widetilde{\widetilde{a}})$ of any noninteger-coordinate point $\widetilde{\widetilde{a}}$ as a weighted average of the 4 pixels in the nearest 2×2 neighborhood of the point (Fig. 4). For any $\widetilde{\widetilde{a}}$, one can easily identify $a_{1,1}$, $a_{1,2}$, $a_{2,1}$, and $a_{2,2}$, which are the four nearest points

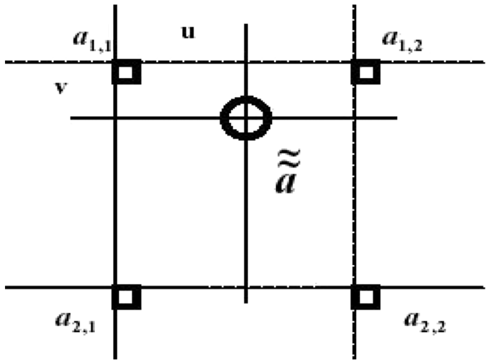


Fig. 4. Bilinear interpolation. We can express the value of a noninteger-coordinate point $\widetilde{\widetilde{a}}$ as a weighted sum of the values of its integer-coordinate neighbors $a_{1,1}, a_{1,2}, a_{2,1}$, and $a_{2,2}$. The point $a_{1,1}$ has a vertical distance v and a horizontal distance u from $\widetilde{\widetilde{a}}$; $a_{1,2}$'s horizontal distance is $(1 - u)$, etc. A pixel's weight coefficient is inversely proportional to its distance from $\widetilde{\widetilde{a}}$, as shown by (1).

with integer coordinates, and the distances u and v indicated in Fig. 4. For any $\widetilde{\widetilde{a}}$, bilinear interpretation approximates that

$$I(\widetilde{\widetilde{a}}) = (1-u)(1-v) \times I(a_{1,1}) + u(1-v) \times I(a_{1,2}) + (1-u)v \times I(a_{2,1}) + uv \times I(a_{2,2}). \quad (1)$$

Thus, for a given noninteger coordinate point $\widetilde{\widetilde{a}}$, each neighboring pixel is assigned a weight coefficient inversely proportional to its distance from $\widetilde{\widetilde{a}}$. Here, $(1-u)(1-v)$ is the weight coefficient of $a_{1,1}$; $u(1-v)$ is the weight coefficient of $a_{1,2}$, etc.

We can now divide the geometric transformation task into two algorithms. 1) A preprocessing algorithm computes the mapping of output pixels onto input pixels and uses bilinear interpolation to relate noninteger-coordinate input points to input pixels. The preprocessing algorithm stores its results in a lookup table wherein each entry relates to a specific output pixel. 2) A real-time algorithm consults the lookup table to quickly calculate the value of an output pixel based on the values of input pixels that map to that output pixel.

For each pixel in the output frame, the preprocessing algorithm works as follows.

1) Convert the output pixel's Cartesian coordinates (x', y') to polar coordinates (r', θ'), where the origin coordinate is the instantaneous center of gaze determined by the tracking algorithm. Map the output pixel's coordinates to an input pixel's coordinates using the inverse mapping function: $(r, \theta) = (f^{-1}(r'), \theta')$ Then, convert the result to Cartesian coordinates: $(x, y) = (r\cos\theta, r\sin\theta)$.
2) Use bilinear interpolation (1) to express the value $I(\widetilde{\widetilde{a}}) \equiv I(x, y)$ as a weighted sum of the values of its integer-coordinate neighbors.
3) Store, in a 2-D lookup table, the set of input pixel coordinates $\{a_{1,1}; a_{1,2}; a_{2,1}; a_{2,2}\}$ and the corresponding set of weight coefficients $\{(1-u)(1-v); u(1-v); (1-u)v; uv\}$ within an entry that is addressed with the output pixel coordinates (x', y').

The lookup table created by this preprocessing algorithm has an entry for each output pixel. Each such entry contains a set of pixel coordinates and a set of weight coefficients. These sets describe how, under a geometric transformation, four appropriate input pixels will influence the value of the output pixel. This preprocessing algorithm does not depend on the actual values of input pixels, since all its operations are on pixel coordinates. Nor does it depend on the direction of gaze, since the pixel coordinates in the lookup table are all relative to the origin coordinate, which is defined as the direction of gaze during a given frame. Therefore, the algorithm may be entirely carried out prior to receiving any input data from the camera.

For a given frame, the real-time algorithm evaluates the intensity $I(x', y')$ of each output pixel (x', y') as follows.

1) Extract, from the output pixel's entry in the lookup table, the set of input pixel coordinates $\{(x, y)_k | k = 1 \dots n\}$ and corresponding weight coefficients $\{w_k | k = 1 \dots n\}$ generated in the preprocessing algorithm.
2) Calculate the output pixel's value $I(x', y')$ with the following weighted sum:

$$I(x', y') = \sum_{k=1}^{n} w_k \cdot I(x, y)_k.$$

Here, $I(x, y)_k$ are the actual values, in the input frame, of the pixels with the coordinates $(x, y)_k$ identified in the preprocessing algorithm.

To generate a lookup table using the coordinates mapping function shown in Fig. 3(B), the preprocessing algorithm executes 10 multiplications, 4 additions, and 3 root operations for each output pixel. All of these operations can be considered as operations "saved" from having to be performed during real-time execution. The real-time algorithm performs only four multiplications and three additions per pixel. These different operations require different amounts of processing power, but we estimate that the preprocessing algorithm confers a gain in efficiency of roughly 50%.

Importantly, the preprocessing algorithm provides a flexible control of image processing. One might expect that individuals with different foveal geometries might require different coordinates mapping functions; new lookup tables reflecting such variations can easily be generated with no change in the implementation of the real-time algorithm. Moreover, the preprocessing algorithm insures that the number of operations per frame during real-time processing is based on the number of output pixels rather than the number of input pixels.

It should be noted that an extrafoveal implant might not require image stretching and, thus, could avoid this step in image processing. However, there are advantages for having central vision and, thus, it is worth exploring the issues related to the associated image processing. In addition, the approach described here based on the preprocessed lookup table provides a general platform for any geometrical transformations of the image in real time for retinal mapping functions, not necessarily limited to a radial foveal spread.

D. Spatial Filtering

A network of connections in the outer plexiform layer performs the first step in the spatial filtering of natural images. In the normal mammalian retina, each horizontal cell in the outer plexiform layer has reciprocal connections with several photoreceptors. A photoreceptor excites a horizontal cell, which then negatively feeds back to that photoreceptor and neighboring photoreceptors, effectively decreasing their signal [22]. A bipolar cell, thus, receives direct input of one sign in the center region of its receptive field, but also feels an opposite-sign influence, mediated by horizontal cells, from surrounding photoreceptors. In this way bipolar cells develop "center-surround" receptive fields [Fig. 5(A)], wherein light at the center of the field has one effect, either excitatory or inhibitory, and light farther away has the opposite effect. Since a uniform pattern of light within a receptive field activates the center and its antagonistic surround approximately equally, yielding much less net bipolar cell stimulation, this center-surround architecture provides a contrast-enhancing spatial filter, sharpening an image and effectively decreasing the amount of information sent to the brain. When photoreceptors are lost in disease, horizontal cell-photoreceptor synapses are lost as well. To effectively replace the outer nuclear layer, therefore, the image processing system must also recreate the spatial filtering produced by these synapses.

In general, a 2-D spatial filter can be defined by its discrete point-spread function (PSF) [21]

$$b_{k,l} = \sum_{n=0}^{N_h-1} \sum_{m=0}^{M_h-1} h_{n,m} a_{k-n,l-m}. \quad (2)$$

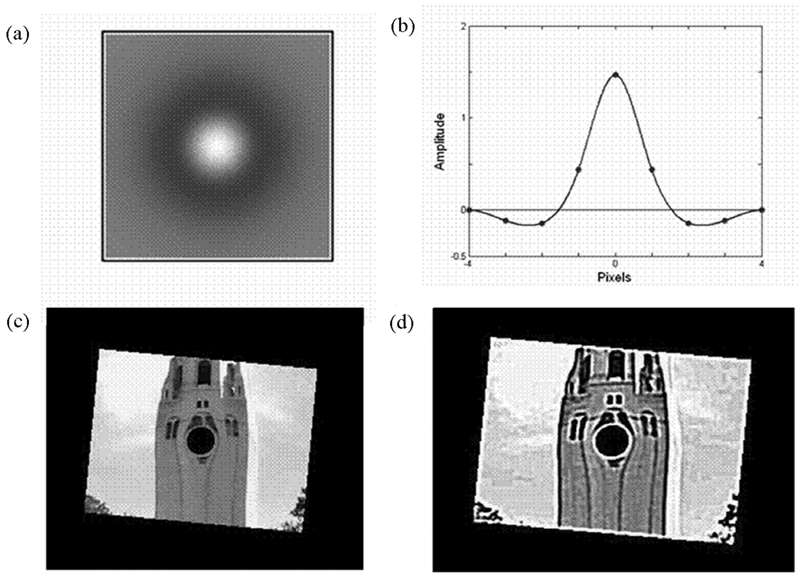


Fig. 5. Spatial filtering. (A) Model of the center-surround receptive field of an ON cell. The cell receives input such that light is excitatory (white) in the receptive field center, inhibitory (dark) in the surround, and neutral (gray) outside the field. (B) A discrete, 7-element, 1-D DOG filter modeling an ON-center receptive field. (C) An image that has undergone cropping and geometrical transformation. (D) The same image with a 2-D DOG filter applied; this particular filter enhances contrast.

Here, $\mathrm{h_{n,m}}$ are the filter coefficients, $\mathrm{a_i}$ is the value of pixel i, and M_h and N_h are the horizontal and vertical extents of the filter.

The required extent of spatial filtering for bipolar cell stimulation can be estimated from published physiological and anatomical measurements. Within 5° of the macaque fovea, ganglion cells with the smallest receptive fields, parvocellular (P) cells, have receptive field centers ranging from 0.05°–0.15° of visual field [23]. Foveal P cells receive excitatory synaptic input from a single midget bipolar cell and, thus, these bipolar cells can be assumed to have receptive field centers of the same size. These receptive field centers are somewhat larger than expected from anatomical data, given that these bipolar and ganglion cells receive direct input from a single cone. The larger size receptive field is thought to arise from electrical coupling between cones. Human P cell receptive fields are likely similar in size, as macaque and human P cells have dendrites of similar size near the fovea [24]. Using a conversion factor of ~300 microns/degree near the fovea, this range corresponds to 1–3 pixels of 14 $\mu\mathrm{m}$ each projected from an LCD display with VGA resolution. Receptive field surrounds for P cells range from 0.16°–1.2° in diameter [23], corresponding to 3–26 LCD pixels. This defines the expected minimum extent for the spatial filter.

The center-surround filters that bipolar cells normally inherit can be modeled as difference-of-Gaussian (DOG) functions [25], combining a positive and a negative Gaussian shape [Fig. 5(B)]. Whereas a single Gaussian filter would smooth an image, a DOG filter sharpens edges and deemphasizes areas of constant intensity Fig. 5(D).

To allow maximum flexibility in the software, each output pixel is assigned its own DOG filter, even though in practice the same DOG filter may be used by many pixels. We first create a lookup table the size of an output frame. Each entry corresponds to an output pixel and contains the discrete PSF values $\mathrm{h_{n,m}}$ of the pixel's assigned DOG filter. During a given frame, we could directly implement each output pixel's filter with the following algorithm.

1) Extract the set of filter coefficients $\{h_{n,m}|n = 0, 1, \ldots N_h - 1; m = 0, 1, \ldots M_h - 1\}$ from the entry addressed by the output pixel's coordinates.
2) Perform spatial filtering by applying (2), where $a_{k,l}$ are the input frame samples.

Such a direct implementation is computationally expensive, requiring $N_h \times M_h$ multiplications and $(N_h - 1) \times (M_h - 1)$ additions per output pixel.

Because a different PSF will be applied to each output pixel in our system, common methods for increasing efficiency, such as recursive filtering and Discrete Fourier Transform Domain filtering, are not appropriate here. Data redundancy is commonly exploited to increase computational efficiency in the compression of images [26]. Due to the high rate of frame acquisition by the scene camera in our system, many input pixel values will not change much between frames; exploiting this data redundancy in the time domain, we can accelerate our image processing software by updating output pixel intensities from their previous values based on changes in input pixel intensities.

To use such an updating paradigm, we must first establish the pixel values of an initial output frame. That is, before we can implement a spatial filtering algorithm capitalizing on data redundancy, we first establish an algorithm to process a single image. To do so, we now combine the geometric transformation step with the spatial filtering step to create a new lookup table structure. We employ the following preprocessing algorithm.

1) Convolve the geometrical transformation lookup table with the spatial filtering lookup table to form a new lookup table. This is possible because both original tables have the same structure—each entry corresponding to one output pixel—and each entry of both tables contains input pixel coordinates and weight coefficients contributing to the value of the entry's output pixel. The result of the convolution is another lookup table with

one entry per output pixel, each entry now containing all the input pixel coordinates and weight coefficients from both the corresponding entries in the original two tables.

2) Generate a new lookup table that is the inverse of the combined table. Each entry in the filter created in step 1) corresponded to an output pixel (x', y') containing the set of input pixel coordinates and weight coefficients contributing to its output value $I(x', y')$. Here, we reorganize that information so that each entry in the new table corresponds to an *input* pixel (x, y) containing the set of output pixels that (x, y) contributes its value to, and the corresponding weight coefficients.

For a given frame, we can now calculate the spatially filtered values of each output pixel with the following real-time algorithm (which we name ISR, for "inverse structure real-time" algorithm):

1) Initialize the value of each output pixel to zero: $I(x', y') = 0$
2) Each entry in the lookup table created in the preprocessing algorithm corresponds to an input pixel (x, y). Extract, from one entry, the set of output pixel coordinates $\{(x', y')_k | k = 1 \ldots n\}$ and the set of weight coefficients $\{w_k | k = 1 \ldots n\}$. For each output pixel $(x', y')_k$ in the extracted set, add the corresponding weight coefficient multiplied by the value of the input pixel (x, y):
   For k = 1 to n, let $I(x', y')_k = I(x', y')_k + w_k I(x, y)$.
   In this step, we distribute the influence of one input pixel to the appropriate output pixels.
3) Repeat step 2) for each entry in the lookup table, distributing the influence of each input pixel.

After the ISR algorithm implements both the geometric transformation and the spatial filtering to calculate output pixels for a given frame, data redundancy is used to calculate pixel values in subsequent frames. As stated, output pixels are updated based on changes in input pixels. Moreover, output pixels will only be updated by increments or decrements that exceed a predefined threshold Φ.

For the data redundancy real-time (DRR) algorithm, the ISR lookup table is first modified so that each entry's set of weight coordinates is sorted in descending order: the set becomes $\{w_k | k = 1 \ldots n\}$, where $w_1 > w_2 > \ldots > w_n$.

The first output frame is computed in one iteration of the real-time algorithm ISR described above. From the second frame on, DRR performs the following for each input pixel (x, y) in the ith input frame:

1) Define $\Delta^i_{(x,y)} = I(x, y)^i - I(x, y)^{i-1}$, the change in input pixel (x, y)'s value between frame i $-$ 1 and frame i.
2) Extract, from the entry addressed by (x, y) in the inverse-logic lookup table, the descending-order set of weight coefficients $\{w_k | k = 1 \ldots n\}$ and their corresponding output pixel coordinates $\{(x', y')_k | k = 1 \ldots n\}$. Define w_j and p_j, initializing their values respectively to w_1 and $(x', y')_1$. If $|w_j \times \Delta^i_{(x,y)}| > \Phi$ then proceed to step 3). Otherwise start step 1) again with the next pixel in the input frame.
3) Add $w_j \times \Delta^i_{(x,y)}$ to the value of pixel p_j. Note that $w_j \times \Delta^i_{(x,y)}$ can be positive or negative.
4) Assign to w_j and p_j the next weight coefficient and the next pixel indices from the sets $\{w_k | k = 1 \ldots n\}$ and $\{(x', y')_k | k = 1 \ldots n\}$ respectively. If $|w_j \times \Delta^i_{(x,y)}| > \Phi$ then return to step 3). Otherwise, start step 1) again with the next pixel in the input frame.

The real-time algorithm DRR computes an output frame by updating the previous output frame. Each input pixel influences the same set of output pixels in this algorithm as it did in ISR, and with the same weights. In DRR, however, it is a variation in an input pixel's intensity that contributes to variations in output pixels' intensities, rather than an absolute value contributing to absolute values as in ISR. Another important difference is that, in DRR, output pixel increments or decrements less than Φ are considered to be negligible and ignored. An input pixel that would influence n output pixels in ISR might influence only r, some number less than n, during a certain frame of DRR. Say that, for input pixel (x, y)

$$\left|\mathrm{w}_1 \times \Delta^i_{(x,y)}\right| > \Phi; \left|\mathrm{w}_2 \times \Delta^i_{(x,y)}\right| > \Phi; \ldots$$
$$\left|\mathrm{w}_{\mathrm{r}-1} \times \Delta^i_{(x,y)}\right| > \Phi;\ \text{but}\ \left|\mathrm{w}_\mathrm{r} \times \Delta^i_{(x,y)}\right| < \Phi.$$

Here, the values of $\{(x', y')_k | k = 1 \ldots r - 1\}$ are updated but $(x', y')_r$'s value is not. Furthermore, the algorithm does not bother checking later output pixels $\{(x', y')_{r+1}, \ldots (x', y')_n\}$ to test whether their increments would be greater than Φ. Because the weight coefficients are arranged in descending order, the fact that $|\mathrm{w_r} \times \Delta^i_{(x,y)}| < \Phi$ means that $|\mathrm{w_j} \times \Delta^i_{(x,y)}| < \Phi$ for $j = r + 1, r + 2, \ldots n$. Thus, none of the output pixels $\{(x', y')_{r+1}, \ldots (x', y')_n\}$ would be changed; after j = r, the input pixel (x, y) will not contribute to any more output pixels and the algorithm can start over with the next input pixel. In this way we save $(n - r)$ multiplication and addition operations per pixel, greatly accelerating the real-time processing.

The more that an input pixel (x, y) changes its value between frames, the higher its $\Delta^i_{(x,y)}$ and, thus, it will contribute its value to more output pixels. Note also that if an input pixel does not change its intensity between two sequential frames, its $\Delta^i_{(x,y)}$ is 0 and the algorithm moves to the next input pixel without performing any further calculations.

Using a spatial filter whose vertical and horizontal extents were both 7 pixels, direct computational implementation performed 53 multiplications and 39 addition steps per pixel to apply a geometric stretch and spatial filter in a video of a natural scene. DRR, operating on the same video with a threshold set at $\Phi = 2$ gray levels, performed on average only 12 multiplications and 12 additions per pixel. Thus, in this particular case, the use of DRR eliminated 41 multiplications and 27 additions per pixel per frame, which we estimate to represent a savings of roughly 75%. In practice, the increase in efficiency conferred by the use of data redundancy will depend on the chosen threshold and the statistics of the visual scene.

A compromise exists in choosing the value of the threshold Φ, between accuracy of the image representation and processing speed. With a higher Φ, the contribution of more input pixels are ignored; therefore, the software performs fewer calculations per frame but output images deviate more from the images that would be calculated from ISR. Moreover, since the data redundancy algorithm is a recursive process, its error is cumulative. A quantification of this tradeoff is discussed in the Results section below.

E. Temporal Filtering

Fluctuations in light intensity cause continuous fluctuations of the bipolar cell membrane potential [27]. At a constant average luminance, bipolar cell responses are captured well by a linear model consisting of the light intensity convolved with a temporal filter [28], [29]. In this model, each photon leads to a change of membrane potential, the impulse response function of cell, which peaks at tens of milliseconds and can last for several hundred milliseconds. This temporal filter is typically biphasic, with the consequence that a step in light intensity causes a transient change in membrane potential. This property emphasizes variations in intensity over steady illumination, although different bipolar cell types vary, ranging from more transient, emphasizing change, to more sustained, emphasizing steady luminance [25], [30].

The temporal filtering properties of a bipolar cell are in part inherited from photoreceptors [28], in part due to neurotransmitter receptor properties of the photoreceptor-bipolar cell synapse [31], and in part from intrinsic membrane dynamics of the bipolar cell [32]. To replace lost filtering properties of the first cell and synapse in the visual pathway, temporal filtering is artificially reintroduced in software.

For a signal defined by its sequence of samples $\{a_i\}$, a discrete temporal filter is applied with the equation

$$b_I = \sum_{i=0}^{I-J} h_i \cdot a_{I-i}. \tag{3}$$

The set of temporal filter coefficients is $\{h_i : i = 0, 1, I - J\}$, where $(I - J + 1)$ is the number of samples included in the filter and I indicates the current sample.

We implement the map of temporal filters as a lookup table with each entry containing an output pixel's temporal filter coefficients. Once again, for the sake of flexibility, we allow each pixel to have its own filter, although in practice many pixels may share the same filter. We carry out the filter for each output pixel by a direct implementation of (3), where $\{a_i : i = J, J{+}1 \ldots I\}$ are the values of the pixel during the last $(\mathrm{I} - \mathrm{J} + 1)$ frames and a_I is the most recent. More specifically, the pixel value a_i, in this case, is the result of the spatial filtering algorithm during frame i; the inputs to the temporal filter have all been cropped, geometrically stretched, and spatially filtered. Note also that the last $(\mathrm{I} - \mathrm{J})$ output frames of the spatial filtering algorithm must constantly be held in storage for use by the temporal filter.

As a DOG spatial filter emphasizes pixels whose values differ from values of neighboring pixels, the temporal filter we use in our software emphasizes pixels whose values change in time. We show an example of a temporal filter's effect in Fig. 6, using a simple, 3-member set of filter coefficients: $\{-(1/2), 1, -(1/2)\}$.

III. Electrical Stimulus Pattern

Normal bipolar cells signal visual information by continuously varying their membrane potential. However, in the case of the electronic implant, visual information is communicated to bipolar cells through a sequence of individual biphasic pulses lasting about 0.5 ms each. A goal of this design is to closely match the expected pattern of signaling to the nervous system. Thus, we considered the most appropriate pattern of stimulation to achieve approximately continuous signaling from bipolar cells given the biophysical mechanisms of neurotransmitter synaptic release.

Extracellular pulses will very briefly depolarize the bipolar cell membrane, and the potential will return nearly to rest at

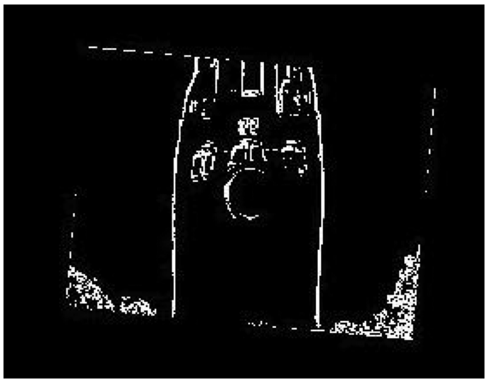


Fig. 6. Temporal filtering. After video images have been cropped, geometrically transformed, and spatially filtered, a temporal filter was applied. The frame shown here is the resulting output after the temporal filter $\{-(1/2), 1, -(1/2)\}$ is applied to a sequence of three sequential images, one which is shown in Fig. 5(D). Light pixels have changed intensity within the previous three frames.

the end of a stimulation pulse. However, bipolar cell synaptic release has ultrafast kinetics in response to changes in membrane potential; depolarizing pulses as brief as 0.5 ms can produce measurable synaptic release, transmitting signals to the next stage of visual processing in the retina. These fast kinetics are limited by the opening of voltage dependent Ca^{++} channels, which have a characteristic time constant of $\sim$1.5 ms [33].

In addition to direct membrane depolarization, extracellular stimulation will have a second effect on the membrane potential, mediated through the resulting ionic currents. Though bipolar cells do not produce action potentials, they do have fast, voltage-dependent Na^{+} channels [34]. In response to the 0.5 ms pulses, it is expected that these channels, plus voltage-dependent Ca^{++} channels, will open, producing a small ionic current which will tend to depolarize the membrane further. This ionic current will have a more lasting effect, being filtered by the membrane time constant of tens of ms [32]. Each pulse will, thus, produce a small signal, acting through ion channels, with kinetics on a millisecond time scale. The pulse amplitude is set by the amount of light from each pixel for each video frame, but the pulse frequency can be set to be greater than the video frame rate, up to 1 kHz. If generating a continuously varying membrane potential were the only goal, a stimulus rate approaching 1 kHz would likely be preferable. However, another important constraint exists: to maintain the level of average power below a safe maximum limit to reduce tissue heating. Thus, an increase in stimulus frequency should be accompanied by a decrease in peak power. As a consequence, a tradeoff exists between the continuous encoding of a signal and the signal's amplitude. Low frequencies would allow higher stimulus amplitude, but would produce a less continuously varying potential. Higher frequencies would allow a more continuous signal at lower amplitude. Perceptually, it is expected that this will translate to a tradeoff between temporal bandwidth and contrast sensitivity. Thus, to optimize the pulse duration and frequency, simple psychophysical tests can be performed, including measuring contrast sensitivity to flashes and drifting gratings at different temporal frequencies. It is also possible that the system may adapt its pulse duration and frequency based on the statistics of the visual input. For a low contrast scene the pulse frequency could be lowered, sacrificing temporal bandwidth in favor of contrast sensitivity,

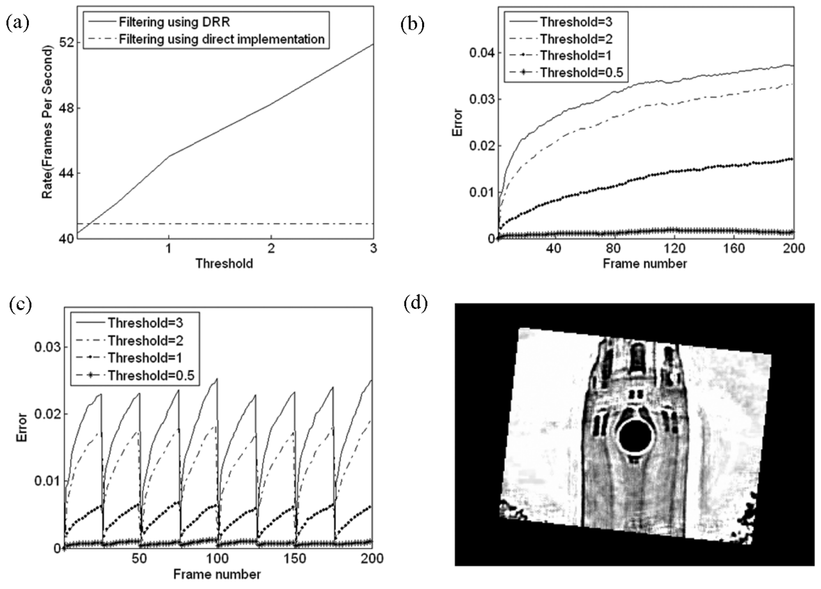


Fig. 7. Software performances measured on our computer and operating on a stored video of a natural scene. (A) Measured maximum frame rates. Using direct implementation of spatial filtering, the software performed image processing at 41 frames per second. When we used DRR instead, the frame rate increased and depended on the threshold Φ (units of gray levels) as shown. (B) Error accumulation with DRR. Using L2 metric [defined by (4)], normalized by the number of pixels in the image, to quantify error, we find that the error accumulates with time and higher thresholds give higher errors. (C) Strategy for reducing error. A use of ISR instead of DRR resets the error to zero; by using ISR once every 25 frames and DRR the rest of the time, we maintain a lower error than in (B) with little sacrifice in efficiency. (D) An image with accumulated error. The input frame is the same as that from Fig. 5(D), and both this image and the image in Fig. 5(d) have been cropped, geometrically transformed, and spatially filtered. However, the spatial filter in Fig. 5(d) was implemented directly, whereas the spatial filter of this frame (and the frames preceding it in stored video) was implemented under the DRR algorithm. The magnitude of this image's error is $\mathrm{L}_2 = 0.03$, quantified (4) by comparison with the errorless spatially filtered image in Fig. 5(D).

whereas for a high contrast scene the pulse frequency could be increased. In fact, contrast adaptation with these general properties is known to occur in the inner retina and higher visual system [28].

Further modifications may provide even more sophisticated temporal patterns of stimulation. To decrease crosstalk between neighboring pixels, the LCD could deliver each output image in a series of interlaced partial frames—this would require additional (but very simple) processing from the controller of the LCD display. Since the IR flash and the power supply of the retinal implant are controlled independently of the LCD screen, no additional image processing power will be required. Pulse duration and repetition rate will be additional adjustable parameters for a patient dialogue-based optimization.

IV. Results and Discussion

We implemented the preprocessing algorithms in Matlab, generating lookup tables in ASCII format; the real-time algorithms were programmed in C. Our software was executed on a computer with an Intel 2.6GHz processor, 512 MB of RAM, under Windows XP operating system. To measure the efficiency of the overall software, we allowed the real-time algorithms to operate at their maximum speed on a stored video—in other words, we timed how long it took various algorithms to process an already-acquired set of consecutive images. (Note that here the only factor limiting the speed of output is algorithm efficiency, whereas when the software is used in an actual prosthesis, its output cannot be generated faster than its 25-Hz input is provided.) The input to the tracking algorithm was a video of three red spots from laser pointers, arranged in a triangle on a dark background to simulate input from reference spots on the implant; this triangle could translate and rotate in time. The input to the cropping, geometrical transformation, spatial filtering, and temporal filtering algorithm was a video of a natural scene that included buildings, trees and moving people. The video was recorded with a hand-held camera, and was similar to what a head-mounted camera would record. When we used a direct computational implementation of filtering, we found that the software could sustain a refresh rate of 41 frames per second (fps) with the tracking and all the image-processing tasks running serially. When we instead used DRR, capitalizing on data redundancy to speed up the stretching and filtering, we achieved higher frame rates; higher DRR thresholds Φ led monotonically to higher frame rates [Fig. 7(A)].

As discussed in the Spatial Filtering section, the use of thresholding introduces an error into the output image [Fig. 7(D)]. Visual information is lost when the algorithm discards the change in an output pixel's value as unimportantly low. Moreover, the error is cumulative in time: under DRR, an output pixel's value may stray increasingly from the value it would have with lossless calculations under ISR. We can quantify an output frame's accumulated error with the distance metric L_2

$$L_2 : \left(\sum_{k=0}^{N-1} |a_k - b_k|^2 \right)^{1/2} \tag{4}$$

where a_k is the value of a pixel k calculated using ISR, b_k is k's value calculated using DRR, and N is the number of pixels in an output frame. Fig. 7(B) shows how error increases with time and how higher thresholds in DRR give faster error accumulation.

Recall that in ISR, there is no error because no data is discarded or ignored; the absolute input pixel intensities determine the absolute output pixel intensities. In DRR, as we adjust output pixel intensities based on changes in input pixel intensities, we must establish the absolute intensities of the first set of output pixels using ISR. To lessen the error accumulation in DRR, the software can periodically "start over", using DRR for most frames but occasionally resetting the error to zero by calculating an output frame with ISR. Fig. 7(C) illustrates how this process zeroes the cumulative error every 25 frames. Here, the frame rate was nearly unchanged from that seen using DRR alone.

Our exploitation of data redundancy allowed us to achieve a refresh rate higher than 50 fps on our system. Further tests will investigate the performance of the software on real portable microcomputers. Rewriting the real-time algorithm in assembly code optimized for the portable microprocessor may further increase to the speed of processing. We believe that our software, run on a Pocket PC, will indeed be able to provide real-time image processing, at the video rate of 25 Hz, for our system.

One might suggest that, if the algorithms using lookup tables are n times as efficient as those using direct implementation, the lookup-table strategy will become unnecessary once pocket PCs' processing power increases by a factor of n. However, it is always prudent for the image processing algorithms to be as efficient as possible: saved computational resources can be devoted to expanding the extents of spatial and temporal filters, increasing the number of transmitted pixels, increasing bit depth, or lowering the DRR threshold Φ.

In an effort to make the software computationally efficient, some aspects of the algorithms have unexpectedly converged with biological solutions in the retina. Because bipolar cells are offset from the foveal center, we likewise shifted visual data away from the fovea through a geometric transformation. This mapping was implemented in software by a fixed lookup table and, thus, could be computed prior to the real-time algorithm. This fixed mapping is analogous to the fixed photoreceptor-bipolar cell synaptic connections, which normally accomplish the mapping between the two coordinate systems. In addition, a biological parallel exists for the DRR algorithm, which measures the change in light intensity between frames within a pixel's spatial receptive field, and then applies a temporal filter to a pixel only if the change exceeds a threshold. This process is very similar to the operation performed by the retina as a whole. Because the bipolar cell temporal filter emphasizes change, a retinal ganglion cell receives a change in synaptic input when the light intensity changes within its receptive field. The ganglion cell then signals with an action potential only when the temporally filtered light intensity exceeds a threshold [35]. The similar strategies have common benefit because both electronic and biological systems face the problem of encoding information through channels of limited dynamic range. In natural visual scenes, changes in light intensity are typically much less common than periods of steady light intensity. Consequently, operations that emphasize temporal change tend to convey visual information more efficiently [36].

The software allows the user to assign separate spatial and temporal filters to every pixel. These parameters include the width and height of the positive and negative Gaussians, and each coefficient of the temporal filter. One strategy for setting these parameters for different regions is based on known physiological properties. For example, bipolar cell dendritic size increases at greater distances from the fovea [37], indicating larger receptive fields requiring spatial filtering across a wider extent. Another reason to allow different regions to contain different filters is that the cell-to-stimulating electrode distances may not be constant over the whole chip. A greater distance would require stronger electrical stimulation to yield a given response in the cell. Varying the amplitude of the filter in different regions could potentially compensate for nonuniformities in electrode-cell distances.

It is important to note that applying different filters to visual information directed to different regions on the implant requires tracking of the implant on that same scale. A more advanced approach could vary filters for individual pixels to address individual bipolar cell types. For example, an increase in light intensity causes ON-type bipolar cells to depolarize, whereas OFF-type bipolar cells hyperpolarize. Applying different filters to the different cell types would require that one electrode will affect primarily one cell, which might be possible with 10 μm electrodes [38], but the feasibility of such a highly resolved stimulation in patients remains to be tested experimentally. The recovery of chromatic sensitivity would similarly require an implant that could address different chromate filters to individual bipolar cells that normally have different chromatic sensitivity. In addition, psychophysical measurements would need to be performed to discover whether an individual bipolar cell transmitted information to the parvocellular pathway, which encodes color information, or the magnocellular pathway, which does not.

A critical component of this system will be the adaptive properties of the nervous system. It is conceivable that upon providing highly resolved chronic electrical stimulation, neurons in the partially degenerated retina will migrate and rewire to adapt and maximally utilize the provided stimulation, but the extent to which this will occur is unknown [13]. However, retinal plasticity in the form of degenerative remodeling or miswiring might also serve as an obstacle to the artificial introduction of visual information [12].

Similarly, another important component of the system will be the adaptability of the software system. To optimize the communication of visual information through the implant, a number of psychophysical tests could be developed to tune the spatio-temporal filters in dialogue with the patient. A direct and quantitative approach to optimize the filters would involve perception of patterns of different spatial and temporal characteristics. Failure to detect or discriminate lines of low or high spatial or temporal frequencies could be directly compensated by a change in the frequency spectrum of the spatial or temporal filters. The software described here gives a great deal of flexibility in image processing and should perform, in clinical trials, at the computational efficiency necessary to allow such investigations.

Acknowledgment

The authors would like to thank Dr. A. Vankov for his help with video recordings and eye movement simulations, and Prof. M. Marmor and Prof. M. Blumenkranz for discussions and encouragement.

References

[1] M. S. Humayun, E. de Juan, Jr., J. D. Weiland, G. Dagnelie, S. Katona, R. Greenberg, and S. Suzuki, "Pattern electrical stimulation of the human retina," *Vision Res.*, vol. 39, pp. 2569–2576, 1999.

[2] S. Y. Kim, S. Sadda, M. S. Humayun, E. de Juan, Jr., B. M. Melia, and W. R. Green, "Morphometric analysis of the macula in eyes with geographic atrophy due to age-related macular degeneration," *Retina*, vol. 22, pp. 464–470, 2002.

[3] S. Y. Kim, S. Sadda, J. Pearlman, M. S. Humayun, E. de Juan, Jr., B. M. Melia, and W. R. Green, "Morphometric analysis of the macula in eyes with disciform age-related macular degeneration," *Retina*, vol. 22, pp. 471–477, 2002.

[4] T. Schanze, M. Wilms, M. Eger, L. Hesse, and R. Eckhorn, "Activation zones in cat visual cortex evoked by electrical retina stimulation," *Graefes Arch. Clin. Exp. Ophthalmol.*, vol. 240, pp. 947–954, 2002.

[5] R. Eckhorn, M. Wilms, T. Schanze, M. Eger, L. Hesse, U. T. Eysel, Z. F. Kisvarday, E. Zrenner, F. Gekeler, H. Schwahn, K. Shinoda, H. Sachs, and P. Walter, "Visual resolution with retinal implants estimated from recordings in cat visual cortex," *Vision Res.*, vol. 46, pp. 2675–2690, 2006.

[6] J. F. Rizzo, 3rd, J. Wyatt, J. Loewenstein, S. Kelly, and D. Shire, "Perceptual efficacy of electrical stimulation of human retina with a microelectrode array during short-term surgical trials," *Invest. Ophthalmol. Vis. Sci.*, vol. 44, pp. 5362–2369, 2003.

[7] J. F. Rizzo, 3rd, J. Wyatt, J. Loewenstein, S. Kelly, and D. Shire, "Methods and perceptual thresholds for short-term electrical stimulation of human retina with microelectrode arrays," *Invest. Ophthalmol. Vis. Sci.*, vol. 44, pp. 5355–5361, 2003.

[8] D. Palanker, A. Vankov, P. Huie, and S. Baccus, "Design of a high-resolution optoelectronic retinal prosthesis," *J. Neural Eng.*, vol. 2, pp. S105–S1020, 2005.

[9] D. Palanker, P. Huie, A. Vankov, A. Asher, and S. Baccus, "Towards high-resolution optoelectronic retinal prosthesis," in *SPIE Proc. BIOS*, 2005, vol. 5688A (Ophthalmic Technologies XV), Paper 37.

[10] M. Becker, M. Braun, and R. Eckmiller, "Retina implant adjustment with reinforcement learning," in *Proc. IEEE Int Conf Acoustics, Speech, and Signal Processing (ICASSP'98)*, Seattle, 1998, vol. II, pp. 1181–1184.

[11] R. Huenermann and R. Eckmiller, "Implementation of tunable receptive field (RF) filters for learning retina implants," in *Proc. ICANN'98*, L. Niklasson, Ed. *et al.*, 1998, pp. 1015–1020.

[12] R. E. Marc, B. W. Jones, C. B. Watt, and E. Strettoi, "Neural remodeling in retinal degeneration," *Prog. Retin. Eye Res.*, vol. 22, pp. 607–655, 2003.

[13] R. E. Marc, personal communication.

[14] P. Azzopardi and A. Cowey, "Preferential representation of the fovea in the primary visual cortex," *Nature*, vol. 361, pp. 719–721, 1993.

[15] D. Palanker, P. Huie, A. Vankov, Y. Freyvert, H. A. Fishman, M. F. Marmor, and M. S. Blumenkranz*, Attracting Retinal Cells to Electrodes for High-Resolution Stimulation*. San Jose, CA: SPIE, 2004, vol. 5314.

[16] D. Palanker, P. Huie, A. Vankov, R. Aramant, M. Seiler, H. Fishman, M. Marmor, and M. Blumenkranz, "Migration of retinal cells through a perforated membrane: Implications for a high-resolution prosthesis," *Invest. Ophthalmol. Vis. Sci.*, vol. 45, pp. 3266–3270, 2004.

[17] K. M. Ahmad, K. Klug, S. Herr, P. Sterling, and S. Schein, "Cell density ratios in a foveal patch in macaque retina," *Vis. Neurosci.*, vol. 20, pp. 189–209, 2003.

[18] A. A. Skavenski, R. M. Hansen, R. M. Steinman, and B. J. Winterson, "Quality of retinal image stabilization during small natural and artificial body rotations in man," *Vision Res.*, vol. 19, pp. 675–683, 1979.

[19] T. T. McMahon, M. Hansen, and M. Viana, "Fixation characteristics in macular disease. Relationship between saccadic frequency, sequencing, and reading rate," *Invest. Ophthalmol. Vis. Sci.*, vol. 32, pp. 567–574, 1991.

[20] S. G. Lisberger, E. J. Morris, and L. Tychsen, "Visual motion processing and sensory-motor integration for smooth pursuit eye movements," *Annu. Rev. Neurosci.*, vol. 10, pp. 97–129, 1987.

[21] L. Yaroslavsky*, Digital Holography and Digital Image Processing*. Boston, MA: Kluwer Academic, 2003.

[22] L. Cadetti and W. B. Thoreson, "Feedback effects of horizontal cell membrane potential on cone calcium currents studied with simultaneous recordings," *J. Neurophysiol.*, vol. 95, pp. 1992–1995, 2006.

[23] L. J. Croner and E. Kaplan, "Receptive fields of P and M ganglion cells across the primate retina," *Vision Res.*, vol. 35, pp. 7–24, 1995.

[24] D. M. Dacey and M. R. Petersen, "Dendritic field size and morphology of midget and parasol ganglion cells of the human retina," *Proc. Nat. Acad. Sci. USA*, vol. 89, pp. 9666–9670, 1992.

[25] D. Dacey, O. S. Packer, L. Diller, D. Brainard, B. Peterson, and B. Lee, "Center surround receptive field structure of cone bipolar cells in primate retina," *Vision Res.*, vol. 40, pp. 1801–1811, 2000.

[26] L. Yaroslavsky and V. Kober, "Redundancy of signals and transformations and computational complexity of signal and image processing," presented at the 12th ICPR, Int. Conf. Pattern Recognition, Jerusalem, Israel, 1994.

[27] F. S. Werblin and J. E. Dowling, "Organization of the retina of the mudpuppy, Necturus maculosus. II. Intracellular recording," *J. Neurophysiol.*, vol. 32, pp. 339–355, 1969.

[28] S. A. Baccus and M. Meister, "Fast and slow contrast adaptation in retinal circuitry," *Neuron*, vol. 36, pp. 909–919, 2002.

[29] F. Rieke, "Temporal contrast adaptation in salamander bipolar cells," *J. Neurosci.*, vol. 21, pp. 9445–9454, 2001.

[30] T. Euler and R. H. Masland, "Light-evoked responses of bipolar cells in a mammalian retina," *J. Neurophysiol.*, vol. 83, pp. 1817–1829, 2000.

[31] S. H. DeVries, "Bipolar cells use kainate and AMPA receptors to filter visual information into separate channels," *Neuron*, vol. 28, pp. 847–856, 2000.

[32] B. Q. Mao, P. R. MacLeish, and J. D. Victor, "Relation between potassium-channel kinetics and the intrinsic dynamics in isolated retinal bipolar cells," *J. Comput. Neurosci.*, vol. 12, pp. 147–163, 2002.

[33] S. Mennerick and G. Matthews, "Ultrafast exocytosis elicited by calcium current in synaptic terminals of retinal bipolar neurons," *Neuron*, vol. 17, pp. 1241–1249, 1996.

[34] D. Zenisek, D. Henry, K. Studholme, S. Yazulla, and G. Matthews, "Voltage-dependent sodium channels are expressed in nonspiking retinal bipolar neurons," *J. Neurosci.*, vol. 21, pp. 4543–4550, 2001.

[35] J. Keat, P. Reinagel, R. C. Reid, and M. Meister, "Predicting every spike: A model for the responses of visual neurons," *Neuron*, vol. 30, pp. 803–817, 2001.

[36] J. H. Van Hateren, "Spatiotemporal contrast sensitivity of early vision," *Vision Res.*, vol. 33, pp. 257–267, 1993.

[37] H. Kolb and D. Marshak, "The midget pathways of the primate retina," *Doc. Ophthalmol.*, vol. 106, pp. 67–81, 2003.

[38] C. Sekirnjak, P. Hottowy, A. Sher, W. Dabrowski, A. M. Litke, and E. J. Chichilnisky, "Electrical stimulation of mammalian retinal ganglion cells with multielectrode arrays," *J. Neurophysiol.*, vol. 95, pp. 3311–3327, 2006.

Alon Asher received the B.Sc. degree in computer science in 1999 from the Technion-Israel Institute of Technology, Haifa, Israel, and the M.Sc. degree in engineering sciences in 2006 from Tel-Aviv University, Tel-Aviv, Israel.

He worked on the retinal prosthesis project in Prof. Daniel Palanker's group at Stanford University, Stanford, CA. Currently, he is an Algorithm Developer at HP invent, Indigo division, Rechvot, Israel. His fields of interests include image processing and real-time algorithms.

William A. Segal received the B.S. degree in physics from Stanford University, Stanford, CA, in 2003. He is working toward the Ph.D. degree in the Stanford Neurosciences Program.

His interests are in circuit-level interactions of prosthetic devices with the nervous system.

Stephen A. Baccus received the B.S. degree in computer science in 1983 and the J.D. degree in 1986, both from the University of Miami, Miami, FL. He received the M.S. degree in computer science from New York University, New York, in 1987, and the Ph.D. degree in neuroscience from the University of Miami in 1998.

He practiced law in Miami, from 1987–1994, and performed postdoctoral research in neuroscience at Harvard University, Cambridge, MA, from 1998–2004. Since 2004, he has been an Assistant Professor in the Department of Neurobiology at the Stanford University School of Medicine. His current research uses experimental and theoretical approaches to understand how the neural circuitry of the retina processes information.

Dr. Baccus received a Stanford University Terman Fellowship in 2004, a Pew Scholar Award in 2005, and a Karl Kirchgessner Vision Science Award in 2005.

Leonid P. Yaroslavsky received the M.S. degree from the Faculty of Radio Engineering, Kharkov Polytechnical Institute, Ukraine in 1961, the Ph.D. degree from the Institute for Information Transmission, Moscow, Russia in 1968, the D.Sc. degree in physics and mathematics from the State Optical Institute, S.-Petersburg, Russia, in 1982.

Until 1995, he headed a Laboratory of Digital Optics at the Institute for Information Transmission Problems, Russian Academy of Sciences. Since 1995, he is a Professor with the Department of Interdisciplinary Studies, Faculty of Engineering, Tel Aviv University, Tel Aviv, Israel. He was also a Visiting Professor at University of Nuernberg-Erlangen, Nuernberg, Germany in 1991, the National Institute of Health, Bethesda, MD in 1992–1995, the Institute of Optics, Orsay, France, in 1995, the Institut Henri Poincare, Paris, France, in 1998, the International Center For Signal Processing, Tampere University of Technology, Tampere, Finland in 1997-2001, 2002, and 2004–2006, Agilent Laboratories, Palo Alto, CA, in 2000, and Gunma University, Kiryu, Japan, in 2003. He is an author of several books and more then 100 papers on digital image processing and digital holography.

Dr. Yaroslavsky is a Fellow of Optical Society of America.

Daniel V. Palanker received the M.S. degree in physics from Armenian State University, Yerevan, Armenia, USSR, and the Ph.D. degree in applied physics (biomedical optics) from the Hebrew University of Jerusalem, Israel in 1984 and 1994, respectively.

He is currently an Assistant Professor in the Department of Ophthalmology, School of Medicine, and Hansen Experimental Physics Laboratory at Stanford University, CA.

His research focuses on interactions of electric field and light with biological cells and tissues, and applications of these interactions to diagnostic, therapeutic and prosthetic technologies. He has authored over 70 scientific publications and two book chapters in the field of biomedical optics and electronics, and received 10 US and international patents.