Real-time Multi-spectral Image Fusion

Tiranee Achalakul
2-106 CST building, Syracuse University
Syracuse, NY 13244
Tel: 315-443-2226, Fax: 315-443-2126 tachalak@syr.edu

Stephen Taylor
2-106 CST building, Syracuse University
Syracuse, NY 13244
Tel: 315-443-2134, steve@scp.syr.edu , www.scp.syr.edu

Abstract

This paper describes a novel real-time multi-spectral imaging capability for surveillance applications. The capability combines a new high-performance multi-spectral camera system with a distributed algorithm that computes a spectral-screening Principal Component Transform (PCT). The camera system uses a novel filter wheel design together with a high-bandwidth CCD camera to allow image cubes to be delivered at 110 frames per second with spectral resolution between 400 and 1000 nm. The filters used in a particular application are selected to highlight a particular object based on its spectral signature. The distributed algorithm allows image streams from a dispersed collection of cameras to be disseminated, viewed, and interpreted by a distributed group of analysts in real-time. It operates on networks of commercial-off-the-shelf multiprocessors connected with high-performance (e.g. gigabit) networking, taking advantage of multi-threading where appropriate. The algorithm uses a concurrent formulation of the PCT to de-correlate and compress a multi-spectral image cube. Spectral screening is used to give features that occur infrequently (e.g. mechanized vehicles in a forest) equal importance to those that occur frequently (e.g. trees in the forest). A human-centered color-mapping scheme is used to maximize the impact of spectral contrast on the human visual system.

To demonstrate the efficacy of the multi-spectral system, plant-life scenes with both real and artificial foliage are used. These scenes demonstrate the systems ability to distinguish elements of a scene, based on spectral contrast, that cannot be distinguished with the naked eye. The capability is evaluated in terms of visual performance, scalability, and real-time throughput. Our previous work on predictive analytical modeling is extended to answer practical design questions such as “For a specified cost, what system can should be constructed and what performance will it attain?”

Keyword: Principal Component Transform, Multi-spectral Camera, Real-time Image Fusion, and Performance Prediction

1. Introduction and Related Work

Multi-spectral camera systems offer the ability to derive surveillance information by looking outside the range of light visible to a human observer, such as the near Ultra-Violate and infrared spectrums. The information in these spectra is compressed into the human visual channels for real-time display to an analyst. Multi-spectral image fusion, the process whereby multiple spectra are compressed into a single color composite image, can be accomplished using a wide variety of techniques that include pixel, feature, and decision level algorithms [Hall 1992]. At the pixel level, raw pixels can be fused using image arithmetic, band-ratio methods [Richards and Jia 1998], wavelet transforms [Li et al. 1995)], maximum contrast selection techniques [Peli et al. 1999], and/or the principal/independent component transforms [Gonzalez and Woods1993, Mackiewicz 1993, Lee 1998]. At the feature level, raw images can be transformed into a representation of objects, such as image segments, shapes, or object orientations [Hall 1992, 1997]. Finally, at the decision level, images can be processed individually and an identity declaration is used to fuse the results [Hall 1992, 1997].

Most of these fusion techniques have been used on a small number of images where they are said to be particularly effective [Richards and Jia 1998, Hall 1992]. The most notable exception is the Principal Component Transform (PCT) which has been employed in a variety of remote sensing applications through post-processing. The PCT [Mackiewicz 1993, Singh 1985&1993] is used to summarize and de-correlate the images by removing redundancy and packing the residual information into a small set of images, termed *principal components*. Unfortunately, PCT is a highly compute intensive algorithm.

Unlike Fourier, Walsh, or Hadamard transforms, the PCT transformation matrix is not separable, and thus, no fast uniprocessor algorithm exists [14].

To prevent the PCT from highlighting only the variation that dominates numerically, it is necessary to augment the algorithm with spectral angle classification [Kruse et al. 1993] prior to the de-correlation process. This has the effect of reducing the importance of an object that occurs frequently in a scene. For example, the spectral signature of a mechanized vehicle embedded in a forest scene will be treated as equally important as the signature associated with trees.

After calculation of the principal components, it is necessary to generate a color-composite image. To achieve this, in collaboration with Mobium Enterprises Inc., we have developed a concurrent human-centered approach [Achalakul 2000a, Boynton 1979, Peterson et al. 1993, Poirson and Wandell, 1993]. This technique attempts to match the spatial-spectral content of the output image with the spatial-spectral processing capabilities of the human visual system.

In our research we are interested in the development of real-time techniques that operate on a wide range of multi- and hyper-spectral image streams concurrently. These include remote sensing streams, with hundreds of spectra, and those based on pre-selected spectral signatures, such as the surveillance technologies described here. Figure 1 shows the general architectural framework we have developed to achieve this form of fusion. It combines a heterogeneous distributed computing platform with real-time multi-/hyper-spectral cameras and other sensors. The computing platform provides the raw computational resources for real-time concurrent fusion algorithms. It is built from a collection of networked PC's, workstations, and shared-memory multiprocessors (SMP's) connected through high-performance gigabit networking. The cameras are connected directly into the dual-PCI-bus of multiprocessors in the network. Multiple sensors are connected at arbitrary points in the network and multiple analysts connect to the resulting image streams using standard web-browsers.

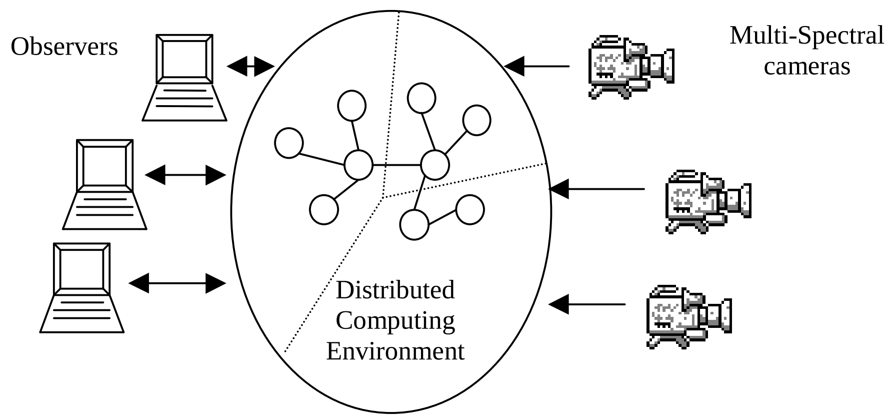


Figure 1: Computing Environment

In collaboration with Mobium Enterprises Inc. and Integrated Scientific Imaging we have developed a novel real-time, multi-spectral camera system for surveillance applications. The system, shown in Figure 2, is capable of acquiring real-time multi-spectral imagery and can be used as a component in the distributed architecture shown in Figure 1. It incorporates a Kodak Megaplus ES1.0 camera capable of delivering 12 bit/pixel images at 55 frames/sec (FPS) at the full resolution of 1024x1024 pixels, or 110 FPS in a 2x2 binning mode at 512x512 pixels. A motorized filter wheel and interchangeable optical system allow 12 independent spectral bands, from 400-1000nm, to be delivered. Designer lenses can be built to suite the optical needs of specific applications can be fitted to the system. The lenses are chosen based on the spectral signatures of the objects under surveillance.

Figure 2: Multi-spectral Camera System

2. Real-time Algorithm

The real-time algorithm is organized using a variant of the manager/worker technique [Chandy and Taylor 1992] depicted in Figure 3. This strategy employ a manager thread that provides the interface to multi-spectral camera, acquires the associated stream of image cubes, and synchronizes the actions of a set of worker threads.

Our previous experimental and analytical studies [Achalakul 2000b] indicate that the PCT algorithm cannot achieve the acquisition rate of the camera system, even with linear speed up, at moderate cost (e.g. $100,000) using today's multiprocessors. However, a significant element of the computation is involved with computation of statistical information that can be reused across subsequent image cubes in an multi-spectral image stream. Thus we chose to utilize two, relatively independent, groups workers: one group performs the primary concurrent PCT algorithm and displays the associated color images in real-time. The other group concurrently computes statistical information and supplies the PCT group whenever possible.

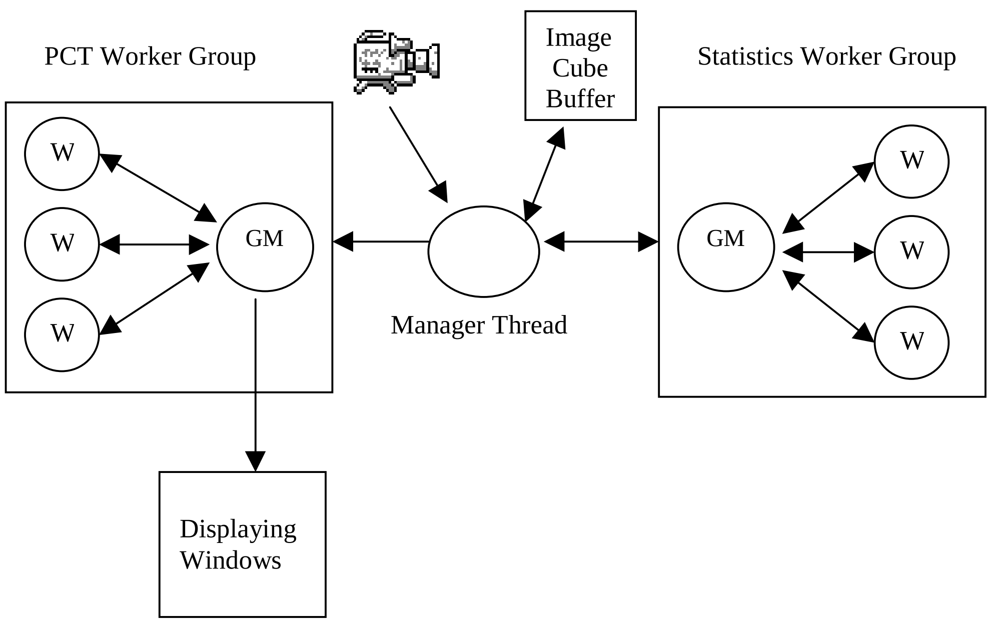
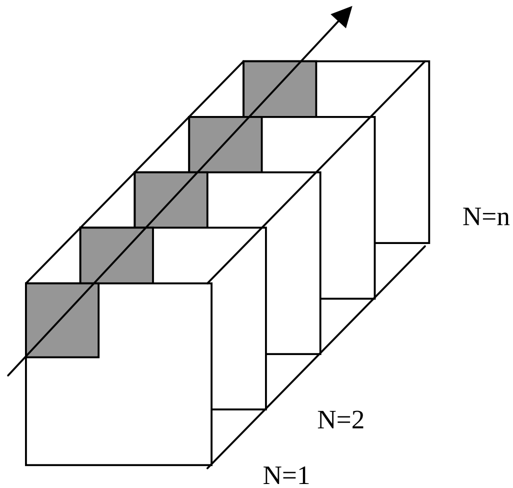


Figure 3: Manager/Worker communication Model

Both worker groups decompose the three-dimensional image-cube structure into sub-cubes that can be operated on independently as shown in Figure 4. This decomposition is achieved through indexing and each sub-cube is operated on by an individual worker thread. Each sub-cube consists of a set of pixel vectors $x_{ij}=[x_1, x_2, \ldots, x_n]$ similar to the decompositions used in [Palmer et al. 1998].



$$x = \begin{bmatrix} x_1 \\ x_2 \\ | \\ x_n \end{bmatrix}$$

Figure 4: Domain Decomposition

The threads may be located anywhere within the distributed computing platform. This yields the abstract code shown in Program 1, which is executed at every multi-processor on the network. The manager and workers are executed as independent threads with a single thread per processor.

```
main() {
   mp = get_my_multiprocessor_id()
   if(mp == 0)
         manager();
   foreach odd multiprocessor
         stats_group();
   foreach even multiprocessor
         pct_group();
}
```

Program 1: Communication Structure

Program 2 shows the abstract code for the manager thread. It allocates a buffer that can hold a preset number of complete image cubes. This buffer is filled by the camera system when cubes are acquired, on a first-in first-out basis (1, 2). The manager also holds the most up to date set of statistics received from the statistics group comprised of a transformation matrix *A* and mean vector *m*. The manager services requests issued by the workers (3). If a worker requests a complete image cube, then the first cube in the FIFO buffer is removed and sent directly (6). If a worker requests the current set of statistics, they are forwarded accordingly (7). Finally, a worker may request to update the current set of statistics when an update is available (8). When there are no requests to be serviced, the manager acquires multi-spectral image cubes from the camera and places them in the buffer (4, 5). When the buffer is full the front cube is discarded, and the entire buffer shifts to make space for the new cube. In this manner, the manager ensures that statistics are applied as soon as they are available, and the PCT operates on the most recent image cube.

The statistical group operates based on the abstract code shown in Program 3. A set of threads is created based on the number of processors allocated to the group (1, 2). An initial request is sent to the manager to obtain the first image cube (3). After this initial request, the processing of each image cube is overlapped with communication of the next cube from the manager (4, 5). The image cube is decomposed using domain decomposition technique depicted in Figure 4 (6). The spectral screening process is applied to each sub-cube concurrently by an individual worker in the group (7). Collectively, the workers in the statistics group build a unique set of pixel vectors *sset* derived from spectral screening across all sub-cubes processed by the group (8). A set of statistics, corresponding to a covariance matrix *cov* and mean vector *m* is then computed based on the pixel vectors (9,10). A transformation matrix *A* is then calculated from the covariance matrix (11). The complete statistics *[A,m]* are then sent to the manager and the process repeats on the next image cube.

```
manager()
{
   cube_buffer = [];                                          //1
   cube_buffer[0] = grab_cube();                              //2
   while (not terminate) {
         request = probe(worker_group);                       //3
         if (no request) {
            cube = grab_cube();                               //4
            cube_buffer = cube_buffer | cube;                 //5
         }
         else {
            if (request == cube)
                  send(worker_group, cube_buffer[0]);         //6
            else if (request == stats && stats available)
                  send(pct_group, [A, m]);                    //7
            else if (request == sendstats)
                  [A, m] = recv (stats_group);                //8
         }
}
```

Program 2: Manager Thread Operation

```
stat_group()
{
   num = get_num_processor();                              //1
   for i = 1 to num
         stats_thr[i] = thread_create();                   //2
   send(manager, req_cube);                                //3
   while( not terminate) {
         cube = recv(manager);                             //4
         send(manager, req_cube);                          //5
         subcube[] = getnerate_subcube(cube);              //6
         for i = 1 to num concurrently
            ssubset[i] = spectral_screening(subcube[i]);   //7
         }
         sset = merge(ssubset[i]);                         //8
         for i = 1 to num concurrently {
            stats[i] = statistics(sset);                   //9
         }
         [cov, m] = merge(stats[i]);                       //10
         A = eigenvectors(cov);                            //11
         send(manager, [A, m]);                            //12
   }
}
```

Program 3: Statistics Worker Group

The PCT worker groups uses the abstract code in Program 4. As with the statistics group, worker threads are created according to the number of processors allocated to the group (1, 2). Two messages are sent to the manager requesting an image cube and the current statistics (3, 4). Once again the algorithm overlaps the computation of the current cube with communication of the next (7, 8). If no new statistics are available, the threads will reuse the previous set for computation. After the image cube is received, it is decomposed (9) and each sub-cube is operated on concurrently by an individual thread to produce a patch of a final color image (10, 11). The patches are eventually accumulated (12) and displayed (13).

```
pct_group()
{
   num = get_num_processor();                                  //1
   for i = 1 to num
         pct_thr[i] = thread_create();                         //2
   send(manager, req_cube);                                    //3
   send(manager, req_stats);                                   //4
   while(not terminate) {
         cube = recv(manager);                                 //5
         [A, m] = recv(manager);                               //6
         send(manager, req_cube);                              //7
         send(manager, req_stats);                             //8
         subcube[] = generate_subcube(cube);                   //9
         for i = 1 to num concurrently {
            pc[i] = PCT(A, m, subcube[i]);                     //10
            RGB[i] = human_centered_mapping(pc[i]);            //11
         }
         image = merge(RGB[i]);                                //12
         display(image);                                       //13
   }
}
```

Program 4: PCT Worker Group

Spectral Angle Classification is a technique that measures the similarity between the spectral signatures of objects in a scene. In a 2-band hyper-spectral space, the similarity between two signatures can be determined by calculating the angle between the two associated pixel vectors X and Y as shown in Figure 5(a). The spectral signatures can then be separated from one another if there is a sufficient difference in their angles as shown in Figure 5(b).

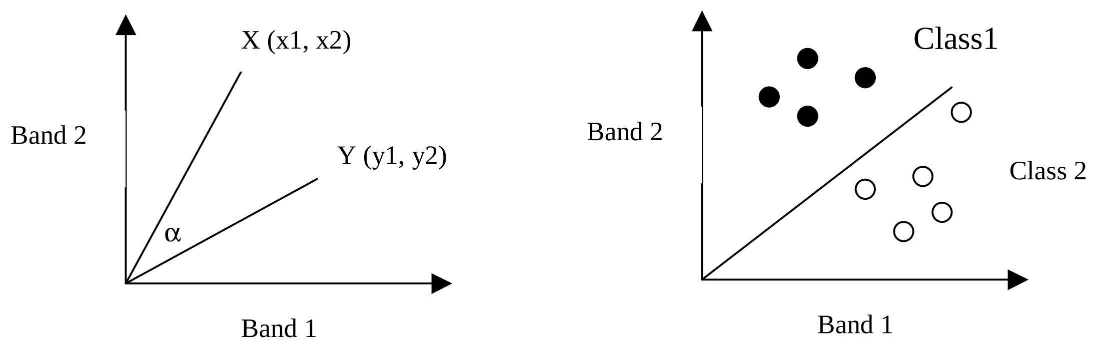


Figure 5: (a) Spectral angle for a two bands image. (b) Classifying spectral space.

Extending this concept from two bands to n-bands, the calculation of the spectral angle can be performed by the following equation that operates on two n-dimensinal pixel vectors.

$$\alpha(x,y) = \cos^{-1}\left(\frac{x \bullet y}{\|x\| \bullet \|y\|}\right) = \cos^{-1}\left(\frac{\sum_{i=1}^{n} x_i y_i}{\left(\sum_{n=1}^{n} x_i^{\ 2} \sum_{n=1}^{n} y_i^{\ 2}\right)^{1/2}}\right)$$

Program 5 shows the abstract code for the spectral screening process. For a given spectral angle threshold, α_{thr}, a set of unique spectral signatures is formed by calculating the spectral angle between all the pixel vectors in a hyper-spectral image using the above equation. The unique signature set is initially empty. Each pixel vector is compared to all of the vectors in the unique set by calculating the associated spectral (2). If all the angles exceed the threshold (3), the pixel vector is added into a set (4); otherwise it is discarded. On completion of the process, a unique set of spectral signatures is determined in which the spectral angle between every pair of pixel vectors is greater than the threshold, α_{thr}. This unique set is then used, instead of the entire collection of pixel vectors in the hyper-spectral image, in the spectral de-correlation process. By adding this screening method, we are assured a variation that dominates numerically (backgound) in the original hyper-spectral image, will not dominate the resulting image; small objects in the scene will have an equal chance of being pushed into the foremost principal components.

```
spectral_screening(cube)
{
   S = {}                                          /* 1*/
   for each v_i in a cube {
         for all v_j in S {
               α(i, j) = cos^-1(i • j / ||i|| • ||j||)   /* 2 */
               if(all (α(i, j)) > α_thr )          /* 3 */
                    S = S U {v_i}                  /* 4 */
         }
   }
}
```

Program 5: Spectral Screening

Principal Component Transform treats each source image as a matrix and forms the associated covariance matrix, which characterizes variations in image contrast. The covariance matrix is then used to form, through a linear transformation, a collection of principal components that effectively summarize the variations across all spectra. The output components carry enough spectral frequency information to reconstruct the original multi-spectral images. The components are rank ordered by the magnitude of their variances (eigenvalues); therefore, most of the spectral contrast is pushed forward to the first few components. The linear transformation thereby permits identification of information that might not be apparent in any single image, or simple linear combination of images that are selected empirically. The PCT algorithm can be divided into two parts that calculate the transformation matrix A, and subsequently transform the data. Consider the pixel vector of the form

$$x = \begin{bmatrix} x_1 \\ x_2 \\ | \\ x_n \end{bmatrix}$$

The mean vector can be defined as

$$m_x = \frac{1}{K}\sum_{k=1}^{K} x_k$$

where K is the number of pixels in an original image set.

The covariance matrix of the n-spectral band image can then be calculated as follows: Because C_x is real and symmetric, finding a set of n orthonormal eigenvectors is always possible [Noble 1969]. The transformation matrix, *A* can then be formed by lining the

$$C_x = \frac{1}{K}\sum_{k=1}^{K} x_k x_k^T - m_x m_x^T$$

sorted eigenvectors calculated from the covariance matrix in each row. The fist row of matrix *A* is the eigenvector corresponding to the largest eigenvalue, and the last row is the eigenvector corresponding to the smallest eigenvalue. Program 6 shows the abstract code for the PCT algorithm in which the multi-spectral image, *I*, is transformed into a set of principal components, *PC*.

```
statistics(sset)
{
        m = 0;
        for  all pixel i in sset
                m = m + i;
        m = m / k;      // where k = number of vectors in sset
        cov = 0;
        for all pixel i in uset {
                C_i = I_i I_i^T - m m^T;
                cov = cov + C_i;
        }
        stats = [cov, m]
}

eigenvectors(stats)
{
        eigvector, eigvalue  = find_eigvector(stats);
        eigvector  =  sort(eigvector, eigvalue)
        A = [eig_1 | eig_2 | … | eig_n]
}

PCT(A, m, cubes)
{
        for all pixel vector V_i in cubes
                PC_i = A(V_i – m);
}
```

Program 6: Principal Component Transform

Human-Centered Color Mapping assigns the first three Principal Components, which have the maximum variance, to a standard representation of the human color space and subsequently converts this representation to RGB values than can be used to drive a color display. A large number of color spaces have been proposed in the literature of color vision [Boynton 1979]. In this paper, we choose to work with the luminance/chrominance model, or YOZ model, favored by Peterson et al. 1993. The response of the three cones in the human visual system can be transformed into a

Luminance band (Y) and two color-opponent bands: red-green (O), and blue-yellow (Z). The information bandwidth of the human color channels is unequal. The spatial frequency bandwidth of the Luminance channel is much greater than the color opponent channels [Poirson and Wandell, 1993]. This suggests that mapping the first Principal Component into the luminance channel and the second and third Principal Components into chromatic channels of the visual system will provide an efficient utilization of the human visual bandwidth.

The YOZ color space is derived from the standard chromacity coordinates termed XYZ, developed by the Commission Internationale de l' Eclairage (CIE) in 1931 using the following empirically derived transform given in Peterson et al. 1993:

$$[YOZ] = [XYZ]\begin{bmatrix} 0 & 0.47 & 0 \\ 1 & -0.37 & 0 \\ 0 & -0.10 & 1 \end{bmatrix}$$

The luminance channel Y is just the CIE Y-channel and the blue-yellow opponent channel Z is just the CIE Z-channel. The red-green opponent channel (O) is given by the equation, O = 0.47X – 0.37Y – 0.10Z.

To obtain the appropriate mapping from YOZ to RGB, we follow the work Boynton. The color space-mapping matrix, k, is derived from the measured spectral power distribution of the display (i.e. intensity at each wavelength) and an empirical color matching function as follows [Boynton 1979]:

$$k = [T\ P]^{-1}$$

The color matching function T is an n-by-3 matrix where each column is determined by having human observers match their color primaries to spectral test lights at different wavelength. Matrix P is a 3-by-n matrix representing the measured spectral power distribution of the primaries. In our experiments, we have used the YOZ color matching functions for matrix T and the spectral power distribution of a typical RGB monitor for matrix P. The normalization of matrix k is shown below:

$$k = \begin{bmatrix} 0.4387 & 0.4972 & 0.0641 \\ 0.4972 & -0.1403 & -0.0795 \\ -0.1355 & 0.0116 & 0.4972 \end{bmatrix}$$

Differential YOZ input values are used because a negative O value indicates green color. The final equation for YOZ to RGB mapping can thus be stated as follows:

$$[RGB] = (128 + (k\_final * ([YOZ] - 128))) / 256$$

3. Example of Image Processing

To demonstrate the real-time multi-spectral capability, we applied it to a simple surveillance task involving the detection of camouflage. Figure 5 shows two scenes taken using a high-quality digital camera. Notice that it is not possible to tell the different between the real and artificial plants from these images.

The multi-spectral camera was aimed around the room containing both real and artificial plants, with the computer controlled filter wheel continuously spinning at the maximal speed of 110 frames per second. Multi-spectral image streams were acquired consisting of 12 gray scale frames, at 512x512 resolution, in the visible and near-infrared spectrum, from 400nm through 1000nm in 50nm steps.

We applied the spectral-screening PCT algorithm to the acquired image stream in real-time. The first principal component, that represents the sum of the overall contrast, contains more than 80% of the total variance. The spectral differences between natural leaves and the artificial material are pushed to the second and third components. There is almost no significant variance left after the third component. Figure 6 shows the first three principal components of the two scenes. Figure 7 shows the resulting color-composite images generating using the human-centered color mapping function presented previously; recall that this maps the first principal component to achromatic, the second to red-green, and the third to blue-yellow opponency. From the resulting images, the artificial plants and flowers are painted blue while the natural plants are painted yellow. There is a clear distinction in color in both experimental scenes. When viewed on a high-quality monitor, the resulting images also show significant variation in the texture detail of the leaves in the scenes.

a) Outdoor Scene

b) Indoor Scene

Figure 5: Scenes from a Digital Camera

Band 1

Band 2

Band 3

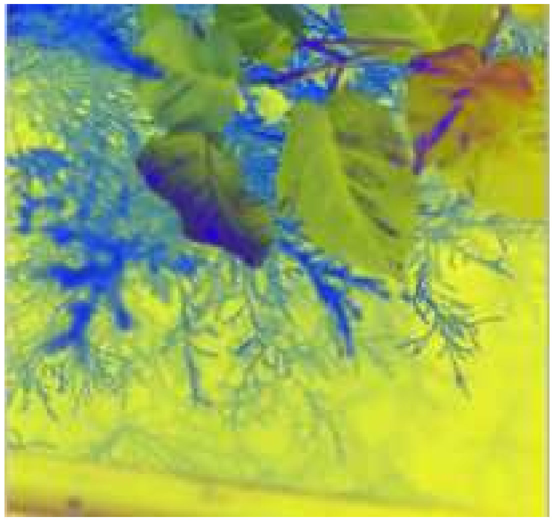
Figure 6a: The First Three Principal Components of the Outdoor Scene

Band 1

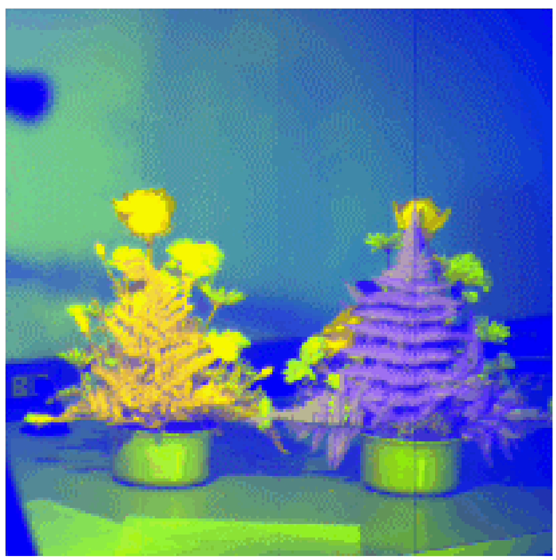
Band 2

Band 3

Figure 6b: The First Three Principal Components of the Indoor Scene



a) Outdoor Scene



b) Indoor Scene

Figure 7: Human-Centered Color Mapping

4. Performance Evaluation and Prediction

It is important to understand the relationship between the statistics and pct worker groups. The real-time performance of the system, i.e. rate of delivery of transformed images, is governed by pct group; however, the speed at which the camera can be panned and still maintain consistent results is governed by the statistics group. These factors can be traded directly, for example, the frame rate can be decreased and pan rate increased by moving the allocation of processors from one group to another. The exact relationship between these rates is *application dependent*. If the scene changes radically from frame to frame, for example – urban environments, then a high rate of statistical update is necessary. However, in applications where the scene is largely uniform, for example - flying over a beach or forest, a low rate of statistical update is acceptible. At the current time there is not a wealth of data to quantify these relationships, however, it is worthwhile to point to the general characteristics via an example experiment.

The performance of the real-time algorithm was measured using two Intel multiprocessors running Windows NT connected through a gigabit network. Each multiprocessor had 8 processors running at 550 Mhz each. The image cube used in the experiment is of 512x512x12 resolution as described in Section 3: Example of Image Processing. Figure 8 shows plots of the speed-up gained as the function of the number of processors, against the ideal speedup, for both the statistics and pct worker groups. Notice that the speedup is close to linear with a performance drop of only 3% in PCT, and 7% in statistics calculations with 8 processors. The performance drop in computing statistics is due to the sequential operations in finding eigenvectors. However, our experience has been that for the problem sizes of interest, the eigenvector calculation is never a dorminant issue.

With 8 computers allocated to each of the statistics and pct worker groups, the statistics can be updated once every 5 seconds, while the pct group is able to produce a frame rate of 3 images/second.

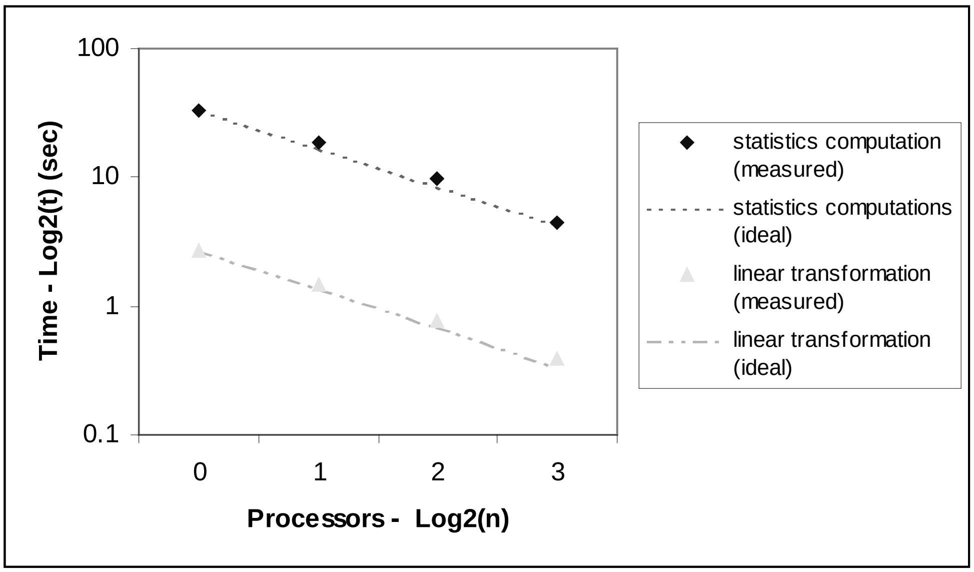


Figure 8. Performance Chart

It can be shown, using techniques developed by Foster et.al. [Foster 1992], that the performace of the spectral-screening PCT can be described analytically with the following predictive model [Achalakul 2000b]:

$$T_t = \frac{k}{p}(C_1 m^2 sn + C_2 sn + C_3 n^2 s + C_4 n^2 m^2 + C_5 m^2) +$$
$$C_6 (p-1)sn + C_7 n^2 p + C_8 n^3 + C_9 T_w \frac{km^2 n}{p} + T_o$$

T_t is the total execution time on an n-band multi-spectral image of size mxm and s is the number of unique spectra in the spectral screening process. T_w represents the network bandwidth, p is the number of processors used, and k represents granularity level. Each step of the algorithm is represented here as a term in the equation that expresses its general computational complexity. The weighting factor C_1 through C_9, and To can be obtained experimentally through a sequence of regression tests and least-square fitting, and the values are listed below.

T_o = 8.8756, C_1 = 7.2833e-009, C_2 = -6.2733e-005, C_3 = -5.2628e-007,

C_4 = 4.1329e-008, C_5 = -4.8906e-005, C_6 = 1.6035e-005, C_7 = -1.6350e-005

C_8 = 8.0959e-006, C_9 = 15.8635

This equation describes the gross behavior of the algorithm when executed on a network of multi-processor system. This analytical model allows the performance of the algorithm to be accurately assessed for changes in image resolution, number of spectral bands, increases in the number of processors, changes in processor technology, networking speeds, and system clock rates; in short, all of the interesting properties of the algorithm on a wide variety of applications. Figure 9 demonstrates the prediction capability of the model. The problem size of 512x512x12 and 1024x1024x12 are used. The algorithm readily scales to 128 processors.

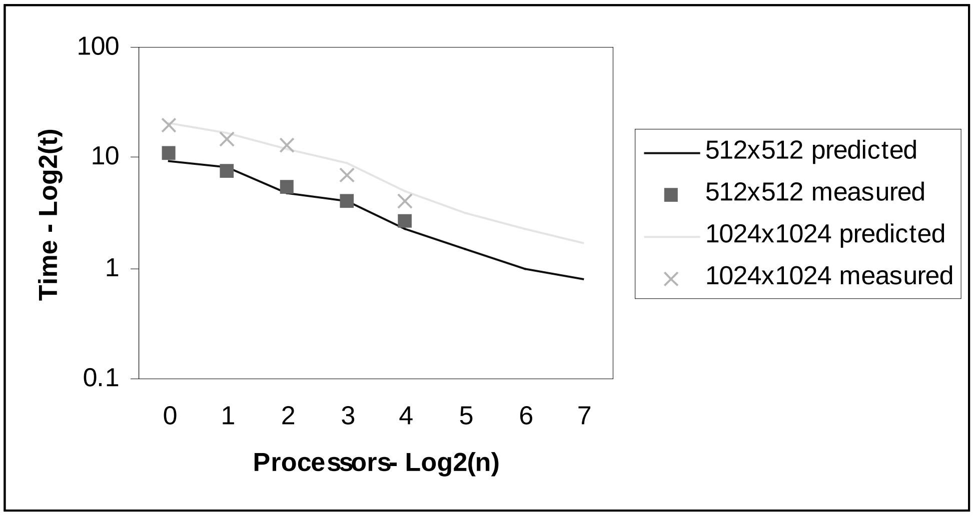


Figure 9: Performance Prediction

Based on this model, with future COTS-multiprocessors containing a modest increase to 16 processors and each processor running at 1GHz, a network of 4 multiprocessors (64 processors), can be expected to produce 30 frames per second at 512x512x12 resolution.

5. Conclusion

This paper described a novel real-time multi-spectral imaging capability and briefly reviewed its algorithms and associated predictive analytical performance model. The system has scalable performance characteristics and currently produces around 3 transformed images per second using an 8-way multiprocessor, updating the statistics infrequently. The images are of 512x512 resolution in 12 separate spectral bands in range from 400 to 1000nm. Modest improvements to the underlying hardware over the next year can be expected to proportionately increase these rates.

The system can be used as a basis to explore a wide variety of applications in which the spectral signatures of the object under surveillance can be exploited with custom designed lenses. For example, detection of mines under a beach, tanks in a forest, contaminants, crop types, and various other land use characteristics. If a particular application requires only two or three filters to accurately detect spectral content, the flywheel can be partially populated and the frame rate proportionately increased.

6. References

1. (a) Achalakul T. and Taylor S., “A Concurrent Spectral-screening PCT Algorithm for Remote Sensing Applications”, Journal of Information Fusion, In press 2000.
2. (b) Achalakul T. and Taylor S., “A Distributed Spectral-Screening PCT Algorithm, submitted to Journal of Software Practice and Experience 2000.
3. Boynton T. M., *Human Color Vision*, Rinehart, and Winston, New York, 1979.
4. Chandy L. M., *Taylor S., An Introduction to Parallel Programming*, Jones and Bartlett publishers, Boston, 1992.
5. Foster I., Gropp W., and Stevens R.. “The Parallel Scalability of the Spectral Transform Method”, Monthly Weather Review, 1992.
6. Gonzalez R. C., Woods R. E., *Digital Image Processing*, Addison-Wesley Publishing Company, Inc., 1993.

7. Hall D.L, "An Introduction to Multisensor Data Fusion," Proceedings of The IEEE, Vol. 85, No. 1, January 1997, pp. 6-23.

8. Hall D.L., *Mathematical Techniques in Multisensor Data Fusion*, Boston, MA:Artech House, 1992.

9. Kruse F. A., Lefkoff A. B., Boardman J. W., Heidebrecht K. B., Shapiro A. T., Barloon P. J., and Goetz F. H., “The spectral Image Processing System (SIPS) – Interactive Visualization and Analysis of Imaging Spectrometer Data”, Remote Sensing Environment vol 44, 1993, pp 145-163.

10. Lee T., *Independent Component Analysis: Theory and Applications*, kluwer Academic Publishers, Boston, 1998.

11. Li H., Manjunath B. A., Mitra S. K., "Multisensor Image Fusion Using the Wavelet Transform," Graphical Models and Image Processing, Vol. 57, No. 3, May 1995, pp. 235-245.

12. Mackiewicz A. and Ratajczak W., “Principal Components Analysis (PCA)”, Computers & Geosciences, vol.19, 1993, pp303-342.

13. Noble B., Apply Linear Algebra, Prentice-Hall, Englewood Cliffs, NJ, 1969.

14. Palmer M., Totty B., Taylor S., “Ray Casting on shared-Memory Architectures: Efficient Exploitation of the Memory Hierarchy”, IEEE Concurrency, Vol 6, No. 1, pp 20-36, 1998.

15. Peli T., Peli E., Ellis K., Stahl R., “Multi-Spectral Image Fusion for Visual Display”, SPIE vol. 3719 Sensor Fusion: Architectures, Algorithms, and Applications III, 1999, pp359-368.

16. Peterson H. A., Ahumada A. J., and Watson A. B., "An Improved Detection Model for DCT Coefficient Quantization", SPIE, vol. 1913, 1993, pp. 191-201.
17. Poirson A. B., Wandell B. A., "Appearance of Colored Patterns: Pattern-color Separability", Journal of the Optical Society of America, vol. 10, 1993, pp 2458-2470.
18. Richards J. A., and Jai X., *Remote Sensing Digital Image Analysis: An Introduction*, New York, NY: Springer, 1998.
19. A. Singh and L. Eklundh, A Comparative Analysis of Standardised and Unstandardised Principal Components Analysis in Remote Sensing, International Journal of Remote Sensing, 14, (1993), 1359-1370.
20. A. Singh and A. Harrison, Standardized Principal Components", International Journal of Remote Sensing, 6, (1985), 883-896.