Dynamic Hyperspectral Imaging

Marina V.A. Murzina*, J. Paul Farrell
Brookhaven Technology Group, 12 Technology Dr., Ste. 12, E. Setauket, NY, USA 11733

ABSTRACT

Bad things often happen fast. This means that we need to react fast. In this work, we develop the technology that allows one to identify and characterize fast events. In real time, we dynamically process hyperspectral information of a scene, specifically analyzing its temporal behavior. The goal is to detect fast and super-fast events like explosions, fast-moving objects and instant changes in the chemical composition of air and other materials.

Until recently, the enormous quantity of hyperspectral information confined us to static hyperspectral data processing. Hyperspectral techniques were used for finding certain objects, chemicals, or anomalies in a picture, frame by frame, statically. Dynamic (temporal) analysis was developed primarily for astrophysical applications performed a long time after the frames had been captured.

In this work, we study ways of taking advantage of emerging hardware technologies that allow one to look at hyperspectral information dynamically: by characterizing temporal changes as they occur. We apply methods from astrophysics (supernova observations) and present our unique algorithms for contemporaneous dynamical analysis of hyperspectral data. The application addresses the question: have there been any sudden changes in the hyperspectral pattern of a scene? If there were sudden changes, were those changes related to a substantial energy release? These questions do not depend on assumptions about specific spectral patterns, chemical composition, or shapes: we look for any changes in a scene. Such dynamical analysis can therefore allow one to react promptly to fast events without prior knowledge about what occurred.

This paper addresses issues specific to dynamic (as opposed to static) hyperspectral imaging, algorithmic approaches to dynamic hyperspectral data processing, and associated hardware-implementation issues.

Keywords: hyperspectral imaging, target tracking, image flow, hyperspectral optical flow, spectro-spatio-temporal analysis, motion detection, autonomous measurements and monitoring, real-time processing, FPGA, DSP, ASIC

1. INTRODUCTION

1.1 Hyperspectral Image Processing

Hyperspectral imaging is a technology which collects and interprets detailed spectral information from a scene. Spectral data for each pixel is represented in dozens or hundreds of narrow, adjacent spectral bands, virtually as a continuous spectrum. The spectral range includes infra-red, visible and ultra-violet light: typically with wavelengths of 400-1000 nm. The resulting images are called hyperspectral cubes, with wavelength (or spectral band number) as “depth”, see fig. 1. The images are then analyzed, allowing one to identify, for example, particular crops, chemicals, geological structures, atmospheric constituents, sick cells in the skin, or hidden armaments.

The main challenge of hyperspectral imaging is the flipside of its advantages: the high volume and complexity of hyperspectral data. In practice, this means that one has to impose significant constraints in order to get meaningful answers in a timely manner. The issue of timely analysis becomes particularly important with the recent development of fast hyperspectral cameras that can provide the information at a rate of up to 30 frames (cubes) per second.
Traditionally, hyperspectral image processing is designed to solve two types of classification tasks:

1) The first is to identify major spectral classes in a scene, the end members. Such classification requires extensive statistical analysis such as Principal Components Analysis or Independent Components Analysis [1-4]. The calculation involves clustering of all data from a given data cube (typically 100 Mbytes in size), and several iterations are often required. This analysis is computationally very expensive and takes a long time. It can also miss small objects [1].

2) The second type of tasks is the classification of hyperspectral data with respect to known reference spectra (signatures). The goal is to determine if specific spectral signatures (e.g., materials) are present in a scene. These calculations can be fast but they require prior knowledge of the target spectra.

Both traditional hyperspectral processing methods analyze one hyperspectral cube at a time, such as a static hyperspectral photograph.

In this work, we introduce an alternative, or a complementary, approach to hyperspectral processing: classification from changes. We consider a sequence of hyperspectral cubes as a hyperspectral movie. The question posed for the analysis is: What parts of a scene are currently significantly changing from frame to frame and what are the changes?

The application can be especially useful for security monitoring, camouflage detection, finding moving objects, analyzing blasts, and finding changing tissues in medical screening.

Because changes can be tracked locally, the calculation does not require extensive statistical analysis and can be very fast. Nor does it require prior knowledge of the spectra of interest. On the contrary, it can learn new spectra for further use. Another advantage of the approach is that it can provide the descriptions (features) of temporal or spatial-temporal behavior of dynamic events. The method can be combined with image reconstruction, i.e., pre-calculating a predicted subsequent image and comparing the actual data with the prediction immediately after new data is acquired.

The ultimate goal is to make hyperspectral change/motion analysis work in real time, recognizing and characterizing dynamic events as they occur.

1.2 Image (Optical) Flow

Technically, dynamic hyperspectral classification can be seen as hyperspectral image flow analysis: a combination of hyperspectral data analysis with the classical image (optical) flow techniques developed for regular movies.

The goal of an optical flow analysis of an image sequence is to find a set of displacement vector fields which relates each image to the subsequent one by locating the same objects in each image, and possibly also by describing changes in a scene. Different methods of optical flow are described in [5, 6, 7]. They include a differential-integral method (optical flow in a narrow sense) which evaluates the overall consistency of brightness flow throughout the scene, gradient-based methods, and a method which tracks feature points (like corners). Traditional optical flow techniques deal with grayscale images. Some methods, like [8], develop techniques for color optical flow. Method [8] makes assumptions about hue conservation, in addition to traditional brightness conservation, of the same object. Because of the additional assumption, the treatment becomes local (as opposed to evaluating the whole image), which makes the calculation fast but not necessarily effective.

Once the same objects in subsequent frames have been identified, one can evaluate changes in a scene. For example, the technique in [9] tracks changes by background subtraction in grayscale images.

Here we propose applying similar techniques to hyperspectral movies. Compared to regular image flow, hyperspectral image flow will allow one to:

- distinguish between objects of similar color and shape based on detailed spectral differences,
- detect camouflaged objects,
- detect events that are mostly seen in the non-visible range of the electromagnetic spectrum (like explosions, warm-blooded animals (including humans), plumes, and exhausts),
- characterize the energy of explosion-type events (whenever it can be deduced from the spectrum),
- save spectral signatures for the new objects for future spectral recognition.
In order to apply the image flow approach to hyperspectral processing, we have to relate frames to each other (in the case of a moving camera), and to characterize changes in a hyperspectral context. In this paper, we assume that the camera is basically still and concentrate on the second task: describing hyperspectral changes of a scene.

In the case of a moving camera, the simplest extension of our approach is to apply standard grayscale optical flow analysis based on the total brightness for frame matching, and then describe hyperspectral changes. A more advanced extension would be using hyperspectral data for finding same objects in sequential data cubes for the case of a moving camera. This can be especially efficient with the feature-point image flow technique, because spectral information helps to identify same feature points in different frames [10].

1.3 Temporal Analysis of Hyperspectral Blast Data

The first attempts at hyperspectral temporal analysis were conducted to collect infrared signatures from bomb detonation and muzzle flashes [11] with the goal of discriminating munition types and sizes. In our research, the goal is the spatio-temporal characterization of dynamic events of different types, without prior knowledge of whether, where, and when an event occurs.

2. HYPERSPECTRAL CLASSIFICATION FROM CHANGES

The goal of our approach is to evaluate changes that occur from one hyperspectral movie frame to another in the related (matched) areas of hyperspectral images. In principle, for an automated analysis, one must also determine appropriate mechanisms for overcoming image noise, and characterize the significant changes in a general enough way for an overall description of the temporal behavior of a scene.

2.1. Spectral Distance Functions

Changes between two sequential hyperspectral frames (cubes) can be evaluated by calculating spectral differences, or spectral distances, between the matching pixels. A spectral distance function can be introduced in a number of ways. In our analysis, we used five possible methods to calculate the distance between the pixel spectrum of one frame \( \bar{b} \) and the next \( \bar{f} \):

1. \( d_{L1} \): linear \( L1 \) distance between the raw spectra:

\[
d_{L1} = \sum_{band=0}^{M} |f_{band} - b_{band}|
\]

which is the area between the two spectral lines, see fig. 2.;

2. \( d_{L1}^N \): linear \( L1 \) distance between spectra that have been normalized to the same maximal value (invariant to brightness);

3. \( d_{L2} \): quadratic \( L2 \) distance between the raw spectra:

\[
d_{L2} = |\bar{f} - \bar{b}| = \left( \sum_{band=0}^{M} \left( f_{band}^2 - b_{band}^2 \right) \right)^{1/2}
\]

4. \( d_{L2}^N \): quadratic \( L2 \) distance between the spectra pre-normalized to maximal brightness;

5. \( d_q \): hierarchical difference between quantized spectra. The procedure is described in section 2.2 below.

These five types of spectral distance function were evaluated in our tests, particularly with respect to their robustness in the presence of noise; see section 4.
2.2. Quantization of Spectral Shapes. Quantized Spectral Distance.

The idea of quantization is to represent spectrum structure in a hierarchical, very compact, way, such as a bit-string. This makes for fast processing and easy data reduction. The quantization procedure involves splitting spectral ranges into segments with more and more refinement, and calculating some attribute for each sub-segment at each level of resolution.

One possible implementation is to detile (i.e., split every segment into two sub-segments) and calculate the area under the spectral curve. In the first step, we calculate two sums: the sum of all band-values of the long-wave part of the spectrum, and the sum for the short-wave side of the spectrum. If the first sum is bigger than the second, the first bit in the output bit-string is 0, otherwise it is 1. Then we divide the short-wave segment into two sub-ranges and calculate a bit which indicates whether or not the short-wave sub-range dominates. We also do this for the long-wave part of the spectrum. In this way, we keep iteratively divide the spectrum into pairs of fragments and hierarchically describe the resulting structure. The process is illustrated below:

![Diagram showing quantization process]

Fig. 3. Schematically shows the quantization technique for hierarchically characterizing spectral structure in the form of a bit-vector. See Table 1 in section 4.1, below.

The first division determines one bit, the second division specifies 2 bits, the third division determines 4 bits, etc. For example, after 5 steps, we calculate \( 1 + 2 + 4 + 8 + 16 = 31 \) bits, and the spectrum is divided into 64 fragments. As a result, we generate just one 31-bit string which roughly describes the spectral structure. This reduces the size of the data approximately 64 times if we consider 128 spectral bands. Quantization boundaries can be optimized for the fastest divergence between given spectra. In section 4, we show actual bit-vectors for actual spectra.

The distance \( d_q \) between quantized spectra can be defined as the level of resolution at which their bit-vectors diverge. The algorithm is exceptionally fast, especially if implemented on an FPGA or ASIC, where direct bit operations are extremely advantageous.
3. DATA

The proposed approach was tested on a fragment of hyperspectral data obtained from [12]. The scene was taken from San Diego State University. Below we show the image as seen in RGB colors and spectral shapes for a few points.

![Hyperspectral data cube used for the background scene: RGB image (center) and spectral shapes for a few points, with light intensity (vertical axes) given in relative scale.](image)

The hyperspectral data cube from fig. 4 was used as a background. In order to introduce dynamic changes in this scene, we imposed an object moving down the road and then exploding. The object has a spectrum varying around the following:

![Characteristic spectrum of the moving object from fig. 4.](image)

The subsequent explosion was introduced as a black-body radiation from a point blast explosion in a uniform ambient medium forming a spherical non-relativistic fireball. The expansion of such a blast follows the classical Sedov-Taylor self-similar solution [13, 14, 15], which is widely used in the physics of strong explosions (e.g., a nuclear bomb) and supernova explosions in astrophysics.

Of course, realistically, the physics of actual explosions is much more complex than our model. Finite duration of the energy release, changing opacity of the fireball and the ambient atmosphere, non-sphericity (particularly, a reverse blast reflected from the ground), atmospheric absorption, realistic energy losses and other effects make the explosion pattern significantly different from our idealistic case. But in addition to its simplicity, the idea of using this approximation is to communicate the possibility of relating hyperspectral data to the physics of the underlying events within the context of real-time hyperspectral processing. In this application, we configure our hyperspectral processing to query whether the data resembles radiation from the self-similar blast and, if so, what are the characteristic parameters of the blast.
Here is a short reminder of the self-similar fireball solution used as a blast example. We expect a spherically-symmetric evolution of the blast wave in space \( r \) and time \( t \). Neglecting the pressure of the external medium, we have only the explosion energy, \( E \), and the external density, \( \rho_0 \), as parameters to the problem. Having only two dimensional variables in addition to radius and time, we can build a non-dimensional combination

\[
\xi = \frac{r}{t^{2/5}} \cdot \left( \frac{\rho_0}{E} \right)^{1/5}
\]  

and rescale the governing hydrodynamic partial-differential equations into a non-dimensional self-similar ordinary differential equation with the variable \( \xi \). This means, in particular, that whenever the flow follows our solution, the character radius, particularly the radius of the expanding shock wave, changes with time as \( t^{2/5} \):

\[
r_{\text{shock}}(t) = \xi_{\text{shock}} \cdot t^{2/5} \cdot \left( \frac{E}{\rho_0} \right)^{1/5}
\]  

where the value \( \xi_{\text{shock}} \) depends on the adiabatic index \( \gamma \). For the atmosphere, \( \gamma = 1.2-1.4 \) depending on ionization. For \( \gamma = 1.2 \), \( \xi_{\text{shock}} = 0.89 \), and \( \gamma = 1.4 \), \( \xi_{\text{shock}} = 1.033 \). A good approximate relation for \( \xi_{\text{shock}}(\gamma) \) is given in [15]. The velocity of the shock wave is then

\[
u_{\text{shock}}(t) = \frac{dr_{\text{shock}}(t)}{dt} = \frac{2}{5} \xi_{\text{shock}} \cdot \frac{1}{t^{1/5}} \cdot \left( \frac{E}{\rho_0} \right)^{1/5}.
\]  

Any character velocity including the speed of sound \( u_{\text{sound}} \) follows the law (5) with some value \( \xi \). So for an ideal gas with \( u_{\text{sound}}^2(t) = \gamma RT / M \) (\( \gamma \) is the adiabatic index, \( R = 8.314 \) J/(mol K) the gas constant, \( M \) is the molecular mass of gas, \( T \) is the temperature in K), the temperature in an opaque fireball follows the relationship

\[
T \propto u_{\text{sound}}^2 \propto \frac{1}{t^{6/5}} \cdot \left( \frac{E}{\rho_0} \right)^{2/5} \propto \frac{1}{r^3} \cdot \frac{E}{\rho_0}.
\]  

with the coefficient of proportionality known from the solution. As soon as the shock front cools down to become transparent to radiation, there forms a characteristic temperature spike right behind the shock. This spike is produced by the ambient gas as it first gets hit and instantaneously heated by the shock, and then quickly radiates its energy through the transparent medium. The relationship for the spike is well known. In our model, we switch to the transparent regime when the optical thickness of the shock front becomes less than 1. The shock was assumed to keep moving according to (4) but the temperature follows the patterns of the transparent shocks waves.

The resulting sequence of images looks roughly as follows:

![Fig. 6. A few frames from the motion-explosion series of pictures. Here we show regular RGB images, but we resolve the whole pattern in the hyperspectral domain.](image-url)
4. RESULTS

Changes in the hyperspectral behavior of a scene-sequence were analyzed by comparing spectral differences between hyperspectral data cubes at sequential moments of time. The spectral difference of each scene pair was evaluated by calculating the distance between the current-frame spectrum of a point and its previous-frame spectrum, for each point. Because we were assuming a still camera, we did not have to match the image points of different frames, so the calculation was performed pixel-by-pixel.

At the same time, we did introduce noise by making neighboring pixels trade their spectra from frame to frame. This method roughly reproduces the natural noise present in the images, as well as a slight movement of the background.

Applying spectral differences to a pair of hyperspectral cubes (frames) produces an image like the picture on the right below:

![Fig. 7. Two sequential hyperspectral images shown in regular color (first and second pictures) and their spectral difference (third picture).](image)

There is actually a slight difference in the background because of noise but it is almost invisible relative to the changes introduced by the moving object.

We used the cube-difference computations to investigate what would be the most efficient (both fast and still providing a satisfactory resolution) spectral distance function to use. The applied distance functions are described in section 2.1. Below we explain the results by considering four spectra shown in fig. 8: two similar spectra from the background, $s_1$ and $s_2$, and two similar spectra from the moving object, $s_3$ and $s_4$. The lack of exactly matching spectra could represent the influence of noise or slight movements in a scene.

![Fig. 8. Two close spectra from the background, $s_1$ and $s_2$, and two close spectra from the moving object, $s_3$ and $s_4.](image)
4.1 Results of Spectral Quantization

First, we will show the results of spectral quantization (section 2.2) applied to spectra from fig. 8.

<table>
<thead>
<tr>
<th>Levels of Resolution:</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Quantized spectra of the Background</strong></td>
<td>bitstring_s1 = 0 - 10 - 1100 - 11100000 - 1110100000000010</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bitstring_s2 = 0 - 10 - 1100 - 11100000 - 1110100000000010</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Quantized spectra of the moving-object</strong></td>
<td>bitstring_s3 = 1 - 10 - 1100 - 11010000 - 1010001000000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bitstring_s4 = 1 - 10 - 1100 - 11010000 - 1010101000000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Bit-vector representation of the spectra from Fig. 8 obtained by the quantization algorithm described in section 2.2. Dashes separate the levels of resolution.

Here the bit-strings are shown by levels of resolution. The distance between spectra was defined as the inverse of the resolution level at which their bit-vectors diverge.

From Table 1, one can see that the bit-strings describing spectral shapes s_3 and s_4 diverge at the last 5-th level of resolution (the sub-bit-strings are different), so their distance is 1/5. The bit-strings for s_1 and s_2 stay same within the resolution applied, so we can say that the distance is 0. At the same time, the spectral shape of the moving object (s_3 or s_4) deviates from the background shapes (s_3 and s_4) already at the first level of resolution; hence their distance is 1, generally showing a large difference between the spectra.

4.2 Spectral Distance Functions and Their Efficiency

In order to evaluate the efficiency of different distance functions, we characterized the noise with respect to the "signal." The noise was characterized as the difference between different background spectra, as well as between the object spectra. The signal was characterized as the difference between the background spectrum and the moving-object spectrum. The ratio of "noise" to "signal" is presented in the last column of Table 2.

<table>
<thead>
<tr>
<th></th>
<th>object - background</th>
<th>object-object</th>
<th>background - background</th>
<th>“noise” / “signal”</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>=</td>
<td>s1 – s3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(d_{L_1})</td>
<td>2956</td>
<td>763</td>
<td>860</td>
<td>0.26</td>
</tr>
<tr>
<td>(d_{L_1}^N)</td>
<td>15.2</td>
<td>5.3</td>
<td>0.93</td>
<td>0.35</td>
</tr>
<tr>
<td>(d_{L_2})</td>
<td>0.1147</td>
<td>0.0073</td>
<td>0.0058</td>
<td>0.0636</td>
</tr>
<tr>
<td>(d_{L_2}^N)</td>
<td>0.0964</td>
<td>0.011</td>
<td>0.0003</td>
<td>0.114</td>
</tr>
<tr>
<td>(d_q)</td>
<td>1</td>
<td>1/5</td>
<td>0</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Table 2. Spectral differences between the spectra from fig. 8 calculated with the five different distance functions described in section 2.2. Differences between the same-nature spectra are considered "noise" and the spectral distance between the background and the moving object is considered "signal". The best signal-to-noise ratio corresponds to the smallest values in the last column.

\[^a\] In principle, two very close shapes, or shape segments, can deviate at inadequately early levels of resolution if they happen to be symmetrical with respect to quantization boundary (the segment borders in fig. 4) and small deviations cause them to “cross the border”. This can cause an instability of quantization and could be overcome by an appropriate choice of boundaries or back-up quantization. This issue requires more research.
From the numbers in the last column, we see that direct Euclidian distance $d_{L2}$ is least sensitive to noise in our example. At the same time, $d_{L1}$ and $d_q$ perform quite efficiently as well, while at the same time being computationally very quick, especially $d_q$. As a whole, all distance functions appeared to be efficient in tracking changes in our example.

4.3 Issues of Hardware Implementation

For hardware implementation, the quantized distance $d_q$ is certainly the most attractive one, as it allows for direct bit-wise operations. After it is optimized for a wide range of data, the method can perform the necessary calculations at speeds much greater than the 30 frames/second needed for timely surveillance, once fast detectors become available.

4.4 Blast Parameters

By finding the areas of spectral change, the blast was easily identified and localized, and its temperature change and the motion of the shock evaluated.

If the atmosphere density $\rho_0$ and the distance to the explosion center are known, and the law of shock motion is evaluated from the hyperspectral images, the equation (4) determines the energy $E$ of the explosion. If the distance to the blast is unknown, hyperspectral observations provide the radius of the blast shock up to a constant factor. The distance can be still found, along with the energy $E$, from equations (4) and (6) and the shock temperature evaluated from the spectrum, as along as the solution is reasonably valid.

As was argued in section 3, in the conditions of real explosion, the gas dynamics of a blast is much more complicated than the solution (3)-(6). Also, in a real observational situation, the calculations of the blast energy spectrum must include atmospheric corrections which might not be trivial if the distance to the object is unknown. In any case, by localizing the areas of spectral changes, one might be able to perform very meaningful hyperspectral analysis in real time extracting basic physical information about an event.

5. CONCLUSIONS

We developed techniques of hyper-spectral image flow analysis which performs hyperspectral classification based on changes between hyperspectral data frames in a hyperspectral movie.

A few approaches to spectral-distance computation were studied. A fast quantization technique for spectral shapes was introduced; this method is particularly efficient if implemented in hardware. We found that any of the distance functions were able to efficiently track hyperspectral changes for prompt detection with the data that we considered.

An example was provided which relates dynamic hyperspectral analysis to the physics of spectrum formation and gas dynamics. The parameters of a blast event were evaluated from spectral data by localizing the areas of spectral change.

This work can be extended in many ways. First, more research is required for testing the application with different realistic hyperspectral movies, for tuning the noise conditions, and for selecting the optimal spectral-distance techniques. Another topic for future research is using hyperspectral data to relate sequential images in a movie in which the camera may move. We expect hyperspectral data to be efficient with methods that register images through feature points like corners. Hyperspectral information could ensure that feature points are identified unambiguously based on their spectra.

For efficient hazard recognition, hyperspectral image flow can be combined with audio data flow like in [16], or more specifically, with acoustic/seismic data like in NORSAR [17].
ACKNOWLEDGEMENTS

We are deeply grateful to our colleague, Dr. D. Usikov, who developed the software for our hyperspectral imaging application and helped us with much scientific advice.

REFERENCES

10. Dr. Daniel Usikov, private conversation.
12. Surface Optics Corporation (SOC), http://www.surfaceoptics.com
17. NORSAR, http://www.norsar.no/NDC/data/