ST. XAVIER'S COLLEGE

MAITIGHAR, KATHMANDU

Real Time System

Theory Assignment #2

Submitted by:

Aashish Raj Shrestha

013BSCCSIT002

Submitted to:

Mr. Ganesh Yogi Lecturer, Department of Computer Science St. Xavier's College	

Date of Submission: 24th August, 2016

Table of Contents

1. Signal Processing 1

1.1. Radar System 1

1.1.1. Tracking 3

1.1.2. Grating 4

1.1.3. Data Association 4

1.2. Other Real Time Applications 6

1.2.1. Real Time Databases 6

1.2.2. Multimedia Application 8

1.2.2.1. MPEG Compression 8

1.2.2.2. MPEG De-compression 9

References 10

1. Signal Processing

A *signal* technically yet generally speaking, is a formal description of a phenomenon evolving over time or space; by signal processing we denote any manual or "mechanical" operation which modifies, analyzes or other-wise manipulates the information contained in a signal [2].

Consider the simple example of ambient temperature: once we have agreed upon a formal model for this physical variable :– Celsius degrees, for instance – we can record the evolution of temperature over time in a variety of ways and the resulting data set represents a temperature "signal". Simple processing operations can then be carried out even just by hand: for example, we can plot the signal on graph paper as in Figure1.1, or we can compute derived parameters such as the average temperature in a month.

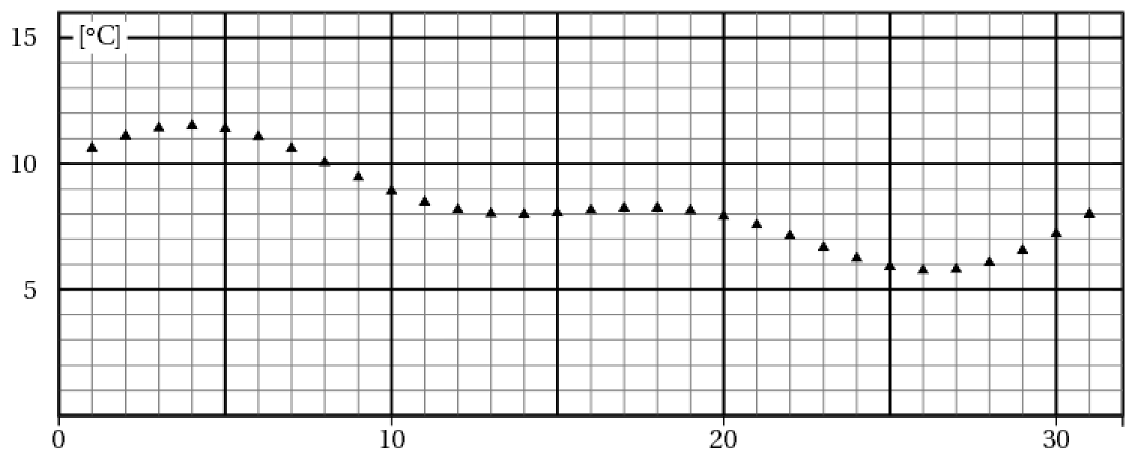

Figure 1.1: Temperature measurements over a month

Conceptually, it is important to note that signal processing operates on an abstract representation of a physical quantity and not on the quantity itself. At the same time, the type of abstract representation we choose for the physical phenomenon of interest determines the nature of a signal processing unit. A temperature regulation device, for instance, is not a signal processing system as a whole. The device does however contain a signal processing core in the feedback control unit which converts the instantaneous measure of the temperature into an ON / OFF trigger for the heating element [2].

The physical nature of this unit depends on the temperature model: a simple design is that of a mechanical device based on the dilation of a metal sensor; more likely, the temperature signal is a voltage generated by a thermocouple and in this case the matched signal processing unit is an operational amplifier.

Most signal processing applications have some kind of real-time requirements. We focus here on those whose response times must be under a few milliseconds to a few seconds. Examples are digital filtering, video and voice compressing/decompression, and radar signal processing [1].

1.1.Radar System

A signal processing application is typically a part of a larger system. As an example, Figure 1–6 shows a block diagram of a (passive) radar signal processing and tracking system. The system consists of an Input / Output (I/O) subsystem that samples and digitizes the echo signal from the radar and places the sampled values in a shared memory. An array of digital signal processors processes these sampled values. The data thus produced are analyzed by one or more data processors, which not only interface with the display system, but also generate commands to control the radar and select parameters to be used by signal processors in the next cycle of data collection and analysis [1].

Radar Signal Processor [3]: The signal processor is that part of the system which separates targets from clutter on the basis of Doppler content and amplitude characteristics. In modern radar sets the conversion of radar signals to digital form is typically accomplished after IF amplification and phase sensitive detection. At this stage they are referred to as video signals, and have a typical bandwidth in the range 250 KHz to 5 MHz. The Sampling Theorem therefore indicates sampling rates between about 500 KHz and 10 MHz. Such rates are well within the capabilities of modern analogue-to-digital converters (ADCs).

The signal processor includes the following components:

- The I&Q Phase Detector,
- The Moving Target Indication (MTI) and
- The Constant False Alarm Rate detection.

The complete proceeding may also be implemented as software in digital receivers.

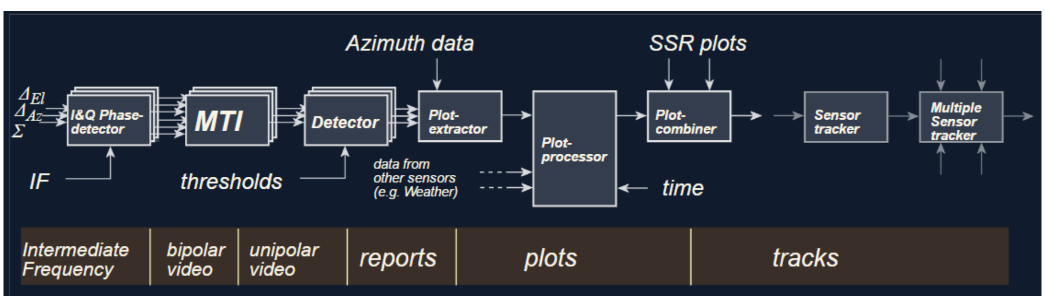

Figure 1.2: Information flow in radar signal processing

The plot extraction and plot processing elements are the final stage in the primary radar sensor chain. The essential process is that of generating and processing plots as distinct from processing waveforms. The main components are:

- The plot extractor or hit processor (translates hits from the signal processor to plots),
- The plot processor (combines primary radar plots and minimizes false plots) and
- The plot combiner (combines primary and secondary plots, uses complementary features to minimize false alarms).

The radar data chain can include the following devices:

- A sensor tracker (it combines some plots of a target to a track), and
- The Multiple Sensor tracker (it combines plots or tracks of other radar sensors).

(The distinction between a correlator and a tracker being, that in the case of a correlator the plot positions are not changed by the process.)

Some of these devises can carried out as a software-module after the digitalizing of the radar data. The Plot Extractor of the Ukrainian company Aerotechnica Corporation (see the picture in Figure 1.3) is a Radar Data Extractors for all types of radars an is designed to upgrade analogue radars.

Figure 1.3: The Plot Extractor A 1000 contains all devices of the radar signal processing. (© Aerotechnica Ltd.)

Radar Signal Processing [1]: To search for objects of interest in its coverage area, the radar scans the area by pointing its antenna in one direction at a time. During the time the antenna dwells in a direction, it first sends a short radio frequency pulse. It then collects and examines the echo signal returning to the antenna.

The echo signal consists solely of background noise if the transmitted pulse does not hit any object. On the other hand, if there is a reflective object (e.g., an airplane or storm cloud) at a distance *x* meters from the antenna, the echo signal reflected by the object returns to the antenna at approximately $2x/c$ seconds after the transmitted pulse, where $c = 3 \times 108$ meters per second is the speed of light.

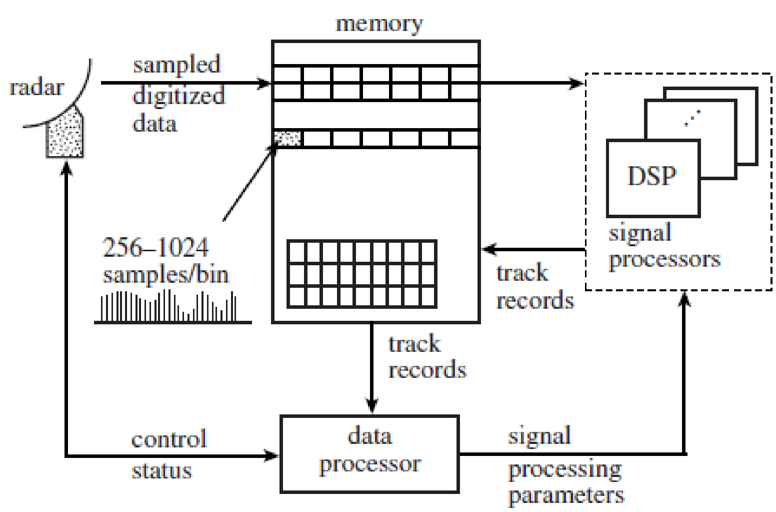

Figure 1.4: Radar Signal Processing and Tracking System

The echo signal collected at this time should be stronger than when there is no reflected signal. If the object is moving, the frequency of the reflected signal is no longer equal to that of the transmitted pulse. The amount of frequency shift (called Doppler shift) is proportional to the velocity of the object. Therefore, by examining the strength and frequency spectrum of the echo signal, the system can determine whether there are objects in the direction pointed at by the antenna and if there are objects, what their positions and velocities are.

Specifically, the system divides the time during which the antenna dwells to collect the echo signal into small disjoint intervals. Each time interval corresponds to a distance range, and the length of the interval is equal to the range resolution divided by *c*. (For example, if the distance resolution is 300 meters, then the range interval is one microsecond long.) The digital sampled values of the echo signal collected during each range interval are placed in a buffer, called a bin in Figure 1–6. The sampled values in each bin are the inputs used by a digital signal processor to produce outputs of the form given by Eq. (1.3). These outputs represent a discrete Fourier transform of the corresponding segment of the echo signal. Based on the characteristics of the transform, the signal processor decides whether there is an object in that distance range. If there is an object, it generates a *track record* containing the position and velocity of the object and places the record in the shared memory.

The time required for signal processing is dominated by the time required to produce the Fourier transforms, and this time is nearly deterministic. The time complexity of Fast Fourier Transform (FFT) is $O(n \log n)$, where n is the number of sampled values in each range bin. n is typically in the range from 128 to a few thousand. So, it takes roughly 103 to 105 multiplications and additions to generate a Fourier transform. Suppose that the antenna dwells in each direction for 100 milliseconds and the range of the radar is divided into 1000 range intervals. Then the signal processing system must do 107 to 109 multiplications and additions per second. This is well within the capability of today's digital signal processors.

However, the 100-millisecond dwell time is a ballpark figure for mechanical radar antennas. This is orders of magnitude larger than that for phase array radars, such as those used in many military applications. A phase array radar can switch the direction of the radar beam electronically, within a millisecond, and may have multiple beams scanning the coverage area and tracking individual objects at the same time. Since the radar can collect data orders of magnitude faster than the rates stated above, the signal processing throughput demand is also considerably higher. This demand is pushing the envelope of digital signal processing technology.

1.1.1. Tracking

Strong noise and man-made interferences, including electronic counter measure (i.e., jamming), can lead the signal processing and detection process to wrong conclusions about the presence of objects. A track record on a non-existing object is called a false return. An application that examines all the track records in order to sort out false returns from real ones and update the trajectories of detected objects is called a *tracker*.10 using the jargon of the subject area, we say that the tracker assigns each measured value (i.e., the tuple of position and velocity contained in each of the track records generated in a scan) to a trajectory. If the trajectory is an existing one, the measured value assigned to it gives the current position and velocity of the object moving along the trajectory. If the trajectory is new, the measured value gives the position and velocity of a possible new object. In the example in Figure 1–6, the tracker runs on one or more data processors which communicate with the signal processors via the shared memory.

1.1.2. Grating

Typically, tracking is carried out in two steps: gating and data association [Bogl]. *Gating* is the process of putting each measured value into one of two categories depending on whether it can or cannot be tentatively assigned to one or more established trajectories. The gating process tentatively assigns a measured value to an established trajectory if it is within a threshold distance G away from the predicted current position and velocity of the object moving along the trajectory. (Below, we call the distance between the measured and predicted values the distance of the assignment.) The threshold G is called the track gate. It is chosen so that the probability of a valid measured value falling in the region bounded by a sphere of radius G centered around a predicted value is a desired constant.

Figure 1.5 illustrates this process. At the start, the tracker computes the predicted position (and velocity) of the object on each established trajectory. In this example, there are two established trajectories, L_1 and L_2. We also call the predicted positions of the objects on these tracks L_1 and L_2. X_1, X_2, and X_3 are the measured values given by three track records. X_1 is assigned to L_1 because it is within distance G from L_1. X_3 is assigned to both L_1 and L_2 for the same reason. On the other hand, X_2 is not assigned to any of the trajectories. It represents either a false return or a new object. Since it is not possible to distinguish between these two cases, the tracker hypothesizes that X_2 is the position of a new object. Subsequent radar data will allow the tracker to either validate or invalidate this hypothesis. In the latter case, the tracker will discard this trajectory from further consideration.

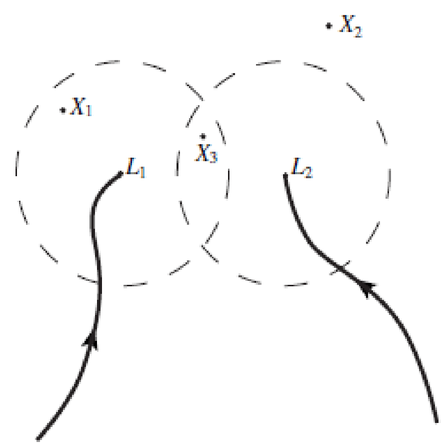

Figure 1.5: Gating process

1.1.3. Data Association

The tracking process completes if, after gating, every measured value is assigned to at most one trajectory and every trajectory is assigned at most one measured value. This is likely to be case when (1) the radar signal is strong and interference is low (and hence false returns are few) and (2) the density of objects is low. Under adverse conditions, the assignment produced by gating may be ambiguous, that is, some measured value is assigned to more than one trajectory or a trajectory is assigned more than one measured value. The data association step is then carried out to complete the assignments and resolve ambiguities.

There are many data association algorithms. One of the most intuitive is the nearest neighbor algorithm. This algorithm works as follows:

1. Examine the tentative assignments produced by the gating step.

a. For each trajectory that is tentatively assigned a single unique measured value, assign the measured value to the trajectory. Discard from further examination the trajectory and the measured value, together with all tentative assignments involving them.

b. For each measured value that is tentatively assigned to a single trajectory, discard the tentative assignments of those measured values that are tentatively assigned to this trajectory if the values are also assigned to some other trajectories.

2. Sort the remaining tentative assignments in order of non-decreasing distance.

3. Assign the measured value given by the first tentative assignment in the list to the corresponding trajectory and discard the measured value and trajectory.

4. Repeat step (3) until the list of tentative assignments is empty. In the example in Figure 1–7, the tentative assignment produced by the gating step is ambiguous. Step (1a) does not eliminate any tentative assignment. However, step (1b) finds that $X1$ is assigned to only $L1$, while $X3$ is assigned to both $L1$ and $L2$. Hence, the assignment of $X3$ to $L1$ is discarded from further consideration. After step (1), there still are two tentative assignments, $X1$ to $L1$ and $X3$ to $L2$. Step (2) leaves them in this order, and the subsequent steps make these assignments. $X2$ initiates a new trajectory. If during subsequent scans, no measured values are assigned to the new trajectory, it will be discarded from further consideration.

The nearest neighbor algorithm attempts to minimize a simple local objective function: the distance (between the measured and predicted values) of each assignment. Data association algorithms of higher time complexity are designed to optimize some global, and therefore more complicated, objective functions, for example, the sum of distances of all assignments and probability of errors. The most complex in both time and space is the class of multiple hypothesis tracking algorithms. Often it is impossible to eliminate some assignments from further consideration by looking at the measured values produced in one scan. (An example is when the distances between measured values to two or more predicted values are essentially equal.) While a single-hypothesis tracking algorithm (e.g., the nearest neighbor algorithm) must choose one assignment from equally good assignments, a multiple-hypothesis tracking algorithm keeps all of them. In other words, a trajectory may be temporally branched into multiple trajectories, each ending at one of many hypothesized current positions. The tracker then uses the data provided in future scans to eliminate some of the branches. The use of this kind of algorithms is confined to where the tracked objects are dense and the number of false returns are large (e.g., for tracking military targets in the presence of decoys and jamming).

1.2. Other Real Time Applications

1.2.1. Real Time Databases

The term real-time database systems refers to a diverse spectrum of information systems, ranging from stock price quotation systems, to track records databases, to real-time file systems.

A **real-time database** is a database system which uses real-time processing to handle workloads whose state is constantly changing. This differs from traditional databases containing persistent data, mostly unaffected by time. For example, a stock market changes very rapidly and is dynamic.

Specifically, a real-time database contains data objects, called *image objects* that represent real-world objects. The attributes of an image object are those of the represented real world object. For example, an air traffic control database contains image objects that represent aircraft in the coverage area. The attributes of such an image object include the position and heading of the aircraft. The values of these attributes are updated periodically based on the measured values of the actual position and heading provided by the radar system. Without this update, the stored position and heading will deviate more and more from the actual position and heading. In this sense, the quality of stored data degrades. This is why we say that real-time data are perishable.

In contrast, an underlying assumption of non-real-time databases (e.g., a payroll database) is that in the absence of updates the data contained in them remain good (i.e., the database remains in some consistent state satisfying all the data integrity constraints of the database).

Real-time databases are useful for accounting, banking, law, medical records, multi-media, process control, reservation systems, and scientific data analysis.

Absolute Temporal Consistency.

The temporal quality of real-time data is often quantified by parameters such as age and temporal dispersion. The age of a data object measures how up-to-date the information provided by the object is. There are many formal definitions of age. Intuitively, the age of an image object at any time is the length of time since the instant of the last update, that is, when its value is made equal to that of the real-world object it represents.11 The age of a data object whose value is computed from the values of other objects is equal to the oldest of the ages of those objects. A set of data objects is said to be absolutely (temporally) consistent if the maximum age of the objects in the set is no greater than a certain threshold of database used to support combat missions of military aircraft. A fighter jet and the targets it tracks move at supersonic speeds. Hence the information on where they are must be less than 50 milliseconds old. On the other hand, an air traffic control system monitors commercial aircraft at subsonic speeds; this is why the absolute temporal consistency threshold for air traffic control is much larger.

Relative Temporal Consistency.

A set of data objects is said to be relatively consistent if the maximum difference in ages of the objects in the set is no greater than the relative consistency threshold used by the application.

For some applications the absolute age of data may not be as important as the differences in their ages. An example is a planning system that correlates traffic densities along a highway with the flow rates of vehicles entering and exiting the highway. The system does not require the most up-to-date flow rates at all interchanges and hence can tolerate a relatively large age (e.g., two minutes). However, if the difference in the ages of flow rates is large (e.g., one minute), the flow rates no longer give a valid snapshot of the traffic scenario and can lead the system to wrong conclusions.

Consistency Models. Concurrency control mechanisms, such as two-phase locking, have traditionally been used to ensure the serializability of read and update transactions and maintain data integrity of non-real-time databases. These mechanisms often make it more difficult for updates to complete in time. Late updates may cause the data to become temporally inconsistent. Yet temporal consistency of real-time data is often as important as, or even more important than, data integrity. For this reason, several weaker consistency models have been proposed (e.g., [KoSp]). Concurrency control mechanisms required to maintain a weaker sense of consistency tend to improve the timeliness of updates and reads.

As an example, we may only require update transactions to be executed in some serializable order. Read-only transactions are not required to be serializable. Some applications may require some stronger consistency (e.g., all real-only transactions perceive the same serialization order of update transactions) while others are satisfied with view consistency (e.g., each read-only transaction perceives some serialization order of update transactions). Usually, the more relaxed the serialization requirement, the more flexibility the system has in interleaving the read and write operations from different transactions, and the easier it is to schedule the transactions and have them complete in time.

Kuo and Mok [Kuo, KuMo93] proposed the use of similarity as a correctness criterion for real-time data. Intuitively, we say that two values of a data object are similar if the difference between the values is within an acceptable threshold from the perspective of every transaction that may read the object. Two views of a transaction are similar if every read operation gets similar values of every data object read by the transaction. Two database states are similar if, in the states, the corresponding values of every data object are similar. Two schedules of a set of transactions are similar if, for any initial state, (1) the transactions transform similar database states to similar final database states and (2) every transaction in the set has similar views in both schedules. Kuo, et al. pointed out that the similarity relation provides a formal means for real-time application developers to capture the semantic constraints of real-time data. They also proposed a concurrent control protocol that takes advantage of the relaxed correctness criterion to enhance the temporal consistency of data.

1.2.2. Multimedia Application

MPEG-1 (Moving Pictures Experts Group) is a video compression and decompression standard [1,2] that achieves roughly the video quality of today's analog TVs and VCRs, at a bit rate of around 1.5 Megabits per second. Studies show that this level of video fidelity is needed for sustained user acceptance. Hence, we chose MPEG-1 over other computationally simpler, but lower fidelity, software-decoded video decompression methods such as Indeo or Quicktime. MPEG-1 defines a compressed bitstream of SIF sized frames (352x240 pixels). MPEG-1 decompression takes such an MPEG-compressed video bitstream, decompresses it, and then renders each decompressed frame. The rest of this section describes MPEG compression and decompression and may be skipped if desired.

1.2.2.1. MPEG Compression

MPEG uses a combination of compression techniques. An intra-coded frame, also called an I-frame, is compressed relative to itself, using only spatial redundancy reduction techniques. JPEG (Joint Photographic Experts Group) type compression for still images [4] is used. A non-intracoded frame uses temporal redundancy (redundancy between frames) as well as spatial redundancy (redundancy within a frame) to reduce the number of bits required to encode the frame. Temporal redundancy tracks the motion of the same objects across frames that are close by in time. Non-intracoded frames are further divided into P-frames and B-frames. P-frames are Predicted frames based on comparisons with an earlier reference frame, while B-frames are Bidirectionally predicted frames, using one backward reference frame and one forward reference frame. A reference frame can be an I-frame or a P-frame, but not a B-frame. By detecting motion of blocks from both a frame that occured earlier and a frame that will be played back later in the video sequence, B-frames can be encoded in even fewer bits. Each frame is divided into macroblocks of 16x16 pixels for the purposes of motion estimation in MPEG compression, and motion compensation in MPEG decompression. A frame with only I-blocks is an I-frame, whereas a P-frame has P-blocks or I-blocks, and a Bframe has B-blocks, P-blocks or I-blocks. For each Pblock in the current frame, the block in the reference frame that matches it best is identified by a motion vector. Then the differences between the pixel values in the matching block in the reference frame and the current block in the current frame is encoded by a Discrete Cosine Transform (DCT). The color space used is the YCbCr color representation, rather than the RGB color space, where Y represents the luminance (or brightness) component, and Cb and Cr represent the chrominance (or color) components. Because human perception is more sensitive to luminance than to chrominance, the Cb and Cr components can be subsampled in both the X and Y dimensions. This means that there is one Cb value, and one Cr value, for every four Y values. Hence, a 16x16 macroblock contains four 8x8 blocks of Y, and only one 8x8 block of Cb and one 8x8 block of Cr values. This is a reduction from the twelve 8x8 blocks (four for each of the three color components), if the Cb and Cr were not subsampled. The six 8x8 blocks in each 16x16 macroblock then undergo transform coding. Transform coding concentrates energy in the lower frequencies. The transformed data values are then quantized, by dividing by the corresponding quantization coefficient. This results in discarding some of the high frequency values, or lower frequency but low energy values, since these become zeros. Both transform coding and quantization enable further compression by runlength encoding of zero values. Finally, the non-zero coefficients of an 8x8 block used in the DCT can be encoded via variable-length entropy encoding, e.g., Huffman coding. Entropy encoding removes coding redundancy by assigning the code words with the fewest number of bits to those coefficients that occur most frequently.

1.2.2.2. MPEG De-compression

MPEG decompression reverses the functional steps taken for MPEG compression. There are 6 basic steps involved in MPEG decompression. First, the MPEG header is decoded. This gives a lot of information about the following block, macroblock, frame or sequence of frames. Second, the compressed video data stream is Huffman decoded from variable-length codes into fixed-length numbers. Next, inverse quantization is performed on the numbers to restore them to their original range. Fourth, an Inverse Discrete Cosine Transform (IDCT) is performed on the 8x8 blocks in each frame. This converts from the frequency domain back to the original spatial domain. This gives the actual pixel values for I-blocks, but only the differences for each pixel for Pblocks and B-blocks. Fifth, motion compensation is performed for P-blocks and B-blocks. The differences calculated in the IDCT step are added to the pixels in the reference block as determined by the motion vector, for P-blocks, and to the average of the forward and backward reference blocks, for B-blocks. In MPEG-1, motion vectors have half-pixel resolution, which again requires taking the average of pixels. Finally, the display step includes color conversion from YCbCr coordinates to RGB color coordinates, and writing to the frame buffer for displaying each frame in the decoded video sequence.

References

[1] Jane W. S. Liu, “Signal processing,” in Real Time System, Ed., 2000, pp. 14-20

[2] P.Prandoni and M. Vetterli, “What Is Digital Signal Processing?,” in Signal Processing for Communications, Ed., 2008, pp. 1

[3] “Radar Signal Processor”, radartutorial.eu, 2016, [Online]. Available: http://www.radartutorial.eu/10.processing/sp05.en.html [Accessed: 24-August-2016].