Comparative Study of Parallelism and Pipelining of RGB to HSL Colour Space Conversion Architecture on FPGA

Phaklen Ehkan, Soon Voon Siew, Fazrul Faiz Zakaria, Mohd Nazri Mohd Warip, Mohd Zaizu Ilyas

Embedded, Networks and Advanced Computing (ENAC) Research Cluster
School of Computer and Communication Engineering, Universiti Malaysia Perlis, Pauh Putra Campus, 02600 Arau, Perlis, Malaysia

Corresponding author’s : phaklen@unimap.edu.my

Abstract. RGB colour model is a basic colour model and complements together to produce full colour range but it is unable to produce sufficient information for digital image analysis. However, HSL is capable to provide other useful information such as colour in degree, saturation of the colour and brightness of colour. In this work, RGB to HSL mathematical conversion algorithm is implemented on FPGA chip. Parallelism and pipelining capabilities of FPGA helps to speed up conversion performance. The RGB to HSL equation is implemented by using two architectures which are parallel and 7-stages pipeline architectures. The designed parallel and pipeline converters have one clock and seven clock cycle of data latency respectively. The parallel and pipeline architectures for RGB to HSL converter have been achieved rate of accuracy by hardware verification up to 99% and 98% and possessed maximum operating frequency merit of 50 MHz and 120 MHz respectively.

Key Words: FPGA, RGB (Red, Green, Blue), HSL (Hue, Saturation, and Luminance), Pipelining, Parallelism.

1. Introduction
Colour space conversion is an important process when performing digital image processing. This conversion process can be performed in either software or hardware domains depending on the processing system. For non-time crucial system, this conversion process commonly implemented in software basis, however for real time system hardware implementation presents a good alternative. The trend of the digital image or digital video resolution is getting higher and higher from Standard Definition (SD) to Ultra High Definition (UHD). This booming trend has directly increased the digital image data volume for processing. While handling such huge data volume, a specific processing core is to be considered during system design in order to reduce the processing time and the output quality.

To date, many existing RGB to HSL converter methods used in image processing but most of them are software-based. The problem with the software is that its implementation is not suitable applied in the real time digital video stream. Besides, the RGB colour model is an additive colour model in which red, green, and blue lights are added up together in numerous ways to reproduce a broad array of colours but there is limitation for RGB colour model to provide sufficient information for image analysis. On the other hand, HSL provides the information such as colour in degree, saturation of the colour and brightness of the colour. These information are much more usable during image analysis. Parallelism and pipelining capabilities of FPGA has simplified the processing steps and only consumes of one clock cycle to produce the HSL value for each pixel without any latency. Hence, the embedded FPGA-based
RGB to HSL converter with less computational time and capable to work on images analysis is developed.

2. Hue, Saturation, and Luminance (HSL) Colour Space

RGB is one of the well-known colour spaces due to its implementation in colour liquid crystal display (LCD) screen, meanwhile HSL usually employed in image processing for specific purpose. Each of the colour space has their own attributes as RGB is categorized as hardware-oriented space while HSL consider human oriented space [1]. HSL is similar with RGB which represented by three components. Hue component can be defined as all the possible colour mixture generated by three primary colour red, green and blue which also defined as rainbow colour [2]. Saturation component is defined as the purity of the particular hue colour and luminance is defined as the amount of light reflected with the particular hue colour [3]. Hue is measured in degree of angle value from 0 to 360 degree. Both saturation and luminance components are measured in percentage from 0% to 100% [2][4-5]. The initial colour of hue is beginning from primary colour pure red to green and green to blue [6]. The generation of hue from primary colour red, green and blue which each has a phase shift of 120 degree to each other in the circular disk with a series of variation each of them from 0 to 360 degrees to produce a rainbow spectrum. Hue and saturation conversion equation is widely available but most of them are for hue, saturation, intensity (HSI) and hue, saturation, value (HSV) colour spaces, but not for HSL colour space. Although the hue conversion equation for all three colour spaces is the same but some works use the equation with trigonometry operation [6-7]. Therefore, those equations are not suitable to implement in this work. Trigonometry operation is difficult and tedious to be implemented in digital architecture, therefore it is not considered. As observed from works by [8-11], they used hue conversion with trigonometry and all worked fine. They implemented this conversion process in software based programming which did not pose any obstacles. The work done by [7] has suggested to adopt the hue conversion equation from [12] which without any trigonometry operation and digital implementation friendly.

This work chooses the RGB to HSL conversion pseudo code from [13]. The conversion steps described by Agoston’s pseudo code did not contain any trigonometry operation. It only involve basic arithmetic add, subtract, multiply and division operation and selection operation. Hence, it is feasible to implement the operation that described by the pseudo code using digital architecture. Equations (1), (2), (3) and (4) are the formularized mathematical RGB to HSL conversion from Agoston’s algorithm where \( r, g, b \) is the 8-bits unsigned binary integer input and \( R, G, B \) is the range of 0 to 1. These equations shown direct attributes helps to reduce the digital architecture design complexity. Before these equations are implemented in digital domain, a simplification process is performed to eliminate some mathematical operations and to normalize the equation to be functional in direct 8-bits unsigned binary input data to avoid floating point operation in the architecture. Floating point architecture is more complex to be implemented and also resources consuming. Fixed point multiplication and division operation is much easy to be implemented by using Look-Up-Table (LUT).

\[
\begin{align*}
(R, G, B) &= \left( \frac{r}{255}, \frac{g}{255}, \frac{b}{255} \right) \\
L &= \frac{\max(R, G, B) + \min(R, G, B)}{2} \\
S &= \begin{cases} 
\frac{\max(R, G, B) - \min(R, G, B)}{\max(R, G, B) + \min(R, G, B)}, & L < 0.5 \\
\frac{\max(R, G, B) - \min(R, G, B)}{2 - \max(R, G, B) - \min(R, G, B)}, & L \geq 0.5 
\end{cases}
\end{align*}
\]
Hue-saturation colour space is a popular choice when comes to the application. Its application covers from object detection and tracking, image segmentation, and also industrial applications. Most of the applications have make use of hue-saturation colour space because of its advantage in representing colour in one dimension (1-D) which is called Hue. Object detection is an important operation in machine vision and real time image processing. This operation is to detect a desired object whether exist in the image or not. Several researches have used hue-saturation colour space for object detection operation [4][10-11][14-16]. Besides, [17-19] applied this colour space to perform image segmentation in their work. [20] had applying hue and saturation colour in agriculture industry to judge the ripeness of the oil palm fruit while [21] has utilizing hue-saturation colour space to perform segmentation in the real time video from abdominal surgery instrument.

3. Implementation of RGB to HSL Conversion Digital Architecture
There are two RGB to HSL conversion architecture (parallel and pipelined architectures) were designed in this work. This design process was performed by using Intel-Altera Quartus II v9.2 and a programming language called Very High Speed Hardware Description Language (VHDL).

3.1 Parallel RGB to HSL Conversion Architecture
As can be observed from the conversion equations (2)-(4), all of them required a maximum, minimum, summation of maximum and minimum, difference of maximum and minimum, and value from the input RGB. These values are then passed to the HSL module to perform HSL conversion operation. Figure 1 shows a hue conversion module which is the most complex module in the parallel converter architecture. Figure 2 displays a saturation conversion module while Figure 3 is the luminance conversion module. The input of RGB value is fixed to 8 bits wide for each component. The hue output value is designed into 10 bits wide while saturation and luminance outputs are 8 bits wide for each of them. In between of this architecture, there is none of register required to store temporary data. All signals are directly fed to the arithmetic operator.

There are several constant values exist in the RGB to HSL conversion operation but do not appeared in the conversion Equations (1), (2), (3) and (4). These constant values are inserted to normalize the whole conversion process and the output value is maintained in integer value. The arithmetic operation before the MUX1 is to prepare the dominator and numerator of the division operation for all three conditions of hue equation in Equation (4). The multiplexer select a correct denominator to be used base on the maximum RGB component. This denominator value is then normalized and fed into a division operator. The second multiplexer MUX2 is used to ensure the final output Hue value is lie within 0 to 360 positive integer ranges. Parallel saturation architecture is less complex if compared to the parallel hue architecture. This saturation conversion architecture is directly implementing according to Equation (3). Both saturation condition are prepared and calculated simultaneously using two division operators. Luminance value then determines which calculated saturation value is being employed via MUX1. The saturation value becomes zero if the input RGB is a grey scale value and achromatic. This is because gray value does not contain any colour information, so the saturation of the colour does not exist. Luminance architecture is the simplest architecture. It constructed by a multiplier and a divider. The summation of maximum and minimum value has been scale up by 10 times and divided with value 51. The value of 51 is produced by the constant 255 from Equation (1) multiplied with the divisor of 255 from Equation (2) and scale down by 10. This parallel architecture serve the purpose to generate the result as fast as possible once the RGB value is supplied to it regardless of the clocking trigger.

\[
H = \begin{cases} 
60 \left( \frac{G - B}{\max(R, G, B) - \min(R, G, B)} \right) & R_{\max} \\
60 \left( 2.0 + \frac{B - R}{\max(R, G, B) - \min(R, G, B)} \right) & G_{\max} \\
60 \left( 4.0 + \frac{R - G}{\max(R, G, B) - \min(R, G, B)} \right) & B_{\max}
\end{cases}
\]
3.2 Pipelined RGB to HSL Conversion Architecture

This RGB to HSL conversion process is divided into 7 stages pipeline process. During development of this pipeline converter, the most complicated Hue equation is broken down into its individual sequential operation and arranges them into 7 stages. It then followed by saturation equation which is also break into sequential steps. Any step that existed in the hue process and shared variable is removed from the saturation equation to avoid redundancy operation and also reduce hardware logic resources. As the result, the saturation also requires a total of seven steps in order to produce the final saturation value. Lastly, the luminance is structured into only five sequential steps. Each step from these three equations is arranged according to the variable dependency in each stage so that all operations are able to process accordingly without any delay. Finally, this pipeline converter contains total of seven stages which are the same as the number of steps in hue equation.

During implementation of hue, saturation and luminance equation in the pipeline, the sign bit is handling using external register. This sign bit is carry through the beginning of the pipeline until the end of the
pipeline. This sign bit is responsible for switching the arithmetic operation for calculating the variable to avoid overflow or negative value occurs. If negative value occurs during the calculation, it will severely disturb the calculation accuracy.

As differ from the parallel architecture, this pipeline architecture requires a clock input pin. This clocking signal used to drive the register in every stage of the pipeline. As depicted in Figure 4, the vertical dashed line represents the registers for each stage. Every intermediate signal and variable produced in each stage of the pipeline is then stored in the register and passing to the next stage. This helps to stabilize the signal before further process.

4. Test Result and Performance Evaluation
A set of standardize testing RGB value is required in order to test and evaluate the designed RGB to HSL conversion architecture. By using random() function in the C code compiler, a total of 1024 sets of random RGB integer values are generated and listed in a text file. This set of random RGB values is then used for C code conversion to generate a reference HSL value and also to test the hardware parallel and converter. During testing process, conversion output accuracy and maximum operating conversion frequency is the main criteria in this work.

4.1 C Code Conversion Algorithm and Data Set Generation
An C code based RGB to HSL conversion based on the Agoston’s algorithm is developed by using Dev-C C code compiler. The purpose is to verify RGB to HSL conversion result and accuracy. After this software based converter is developed, it is used to convert the 1024 sets of predefine random generated RGB values to obtain the 1024 sets of HSL values as a reference for later process of conversion accuracy evaluation. All the C code converted 1024 sets of HSL values are defined into 2 decimal places for accuracy.

4.2 Simulation Results of Parallel and Pipeline Converter
This section is taking both of the converters for functional simulation verification. The main purpose for this simulation test is to verify the theoretical functional performance for both converters. Both simulations use the same set of RGB input value as the C code reference RGB input value. However, only the first 50 sets of RGB value are used in both simulations. Figure 5 represents the simulation outcome for parallel architecture RGB to HSL converter. It clearly shows that the HSL value is produced instantly without any delay when the RGB value is given. This simulation phenomenon has proved its main parallelism attribute.

![Figure 5](image-url)

For the pipeline simulation, there are two additional signals are needed. The first signal is the clock signal labeled as clock_in, and a second signal is the reset signal. The clock_in signal is used to drive the registers in each stage. Since the developed pipeline converter has seven stages, this means that the pipeline has seven clock cycles of data latency. The data latency has proved in the simulation result as shown in Figure 6. The first HSL result is produced after seven clock cycles since the input of RGB value. The HSL output data before the seventh clock cycle is considered invalid. As comparison for both simulation results, it is noticeable that both parallel and pipeline architecture converters are capable to produce the similar HSL output value. Therefore, both of the converters architecture is working properly.
4.3 Hardware Conversion Testing Architecture

To perform this pure hardware conversion testing, a specific testing architecture is developed specifically for RGB to HSL conversion process. Figure 7 is the hardware testing architecture developed in this work. It consists of five 8bit x 1024 word embedded Cyclone II M4K RAM and one 9bit x 1024word M4K RAM. The first three RAM is used to store the C code predefine 1024 sets of RGB values. The remaining two 8-bit and 9-bit RAM are used to store the converted HSL output values. All 1024 sets of predefine RGB values are written into RAM during initialization of the test architecture in FPGA.

![Figure 7. Hardware testing architecture.](image)

After all 1024 sets of data are done with conversion; all the HSL output values stored in the RAM are then read and transmitted to the host computer by using RS232 serial communication protocol. The RS232 controller firstly convert the HSL value in binary into ASCII format before transmit to host computer. The test circuit control unit is responsible to handle and signal the conversion process step and other functional module. This testing architecture is also implemented with a Phase-Lock-Loop (PLL). This PLL is important and used to generate difference clock speed to drive this testing architecture and also the control unit. It used to investigate both parallel and pipeline converter output accuracy in respond to difference clock speed and to investigate their maximum operating frequency.

4.4 Slow Speed Conversion Accuracy

Before both converters are undergoing variable operating frequency testing, both of them had gone thru a slow speed test which is running at 10Hz clock. This test is to ensure that both converters are capable to produce the correct HSL output value when implementing in hardware. This output result also used as reference result when both converters are tested at high clock speed. Table 1 represents the accuracy test for 1024 RGB samples at the data rate of 10Hz clock.

| Converter | Accuracy  |
|-----------|-----------|
| Parallel  | 99.00%    |
| Pipeline  | 98.96%    |

![Table 1. Accuracy of 10Hz data rate for 1024 RGB samples.](image)
4.5 High Speed HSL Conversion Accuracy between of Parallel and Pipeline Architecture

The clock frequency used begins from 10MHz up to 150MHz. It is generated by the PLL. The maximum test frequency ceiling at 150MHz because it has fulfil the High Definition (HD) display clock speed. The accuracy of both converters is compared and analysed by individual output component of hue, saturation and luminance. This is because each of hue, saturation and luminance output values are produced by individual module; therefore, each of component accuracy is different.

Figure 8 shows comparisons of hue conversion accuracy. Both converters are capable to sustain at 99.20% accuracy at the beginning of the test until 50MHz. At 55MHz, the hue accuracy of parallel converter begins to fall. It shows that hue component from parallel converter has a maximum operating frequency at 50MHz. When both converters are tested with 70MHz clock, the parallel converter hue output accuracy start to fall to 19.38% which can considered as useless while the pipeline still able to maintain 99.07% accuracy. The pipeline hue conversion is capable to sustain above 90% up to 150MHz.

![Figure 8. Hue conversion accuracy comparisons.](image1)

From Figure 9, the parallel saturation conversion accuracy sustains at 98.97% started from 10MHz until 70MHz. When frequency is pushed up to 85MHz, the accuracy for parallel architecture dropped below level 80%. The parallel saturation conversion accuracy started to fall at the operating frequency of 75MHz. The pipeline saturation conversion accuracy is sustaining until 85MHz. When the frequency reaches 90MHz, the accuracy for pipeline architecture begin to drop. Significantly, when the frequency is pushed up to 150MHz, the pipeline converter still maintains the accuracy level of 45% and -40% for the parallel architecture.

![Figure 9. Saturation conversion accuracy comparisons.](image2)

Luminance conversion posed the most stable conversion output value. This can be seen in Figure 10. Its accuracy is able to sustain at level of 98.72 % started from frequency 10MHz until 80MHz. But the pipeline luminance is able to keep the accuracy until 100MHz, which is 20MHz higher than parallel converter. Parallel luminance conversion accuracy starts falling when the frequency reaches 85MHz. When frequency goes beyond 120MHz, the accuracy drops below 80% at which the output of the converter is no more reliable and useless. With the comparison to the pipelined architecture, the accuracy is capable to sustain at 96.45% up to 110MHz which 30MHz higher than the parallel architecture. The

![Figure 10. Luminance conversion accuracy comparisons.](image3)
accuracy of the pipelined converter begin to fall when frequency pushed up beyond 110MHz. Significantly, when the frequency tested until 150MHz, the pipeline converter still maintain the accuracy level of 92% compared to 65.4% for the parallel architecture.

It is obviously that the pipeline hue conversion able to withstand high operating frequency up to 120MHz but the parallel converter only can properly function up to 50MHz, which is only half of the pipeline converter.

4.6 Real time Hardware Implementation

Both converters are implemented in the DE2_TV sample project provided by Altera to perform real time conversion operation. DE2_TV sample project is interfaced with a Close Circuit Television (CCTV) imaging sensor to capture real time digital video stream. The video stream is converted into RGB value to be displayed in VGA screen. By making use of this RGB stream, both tested RGB to HSL converters are implemented into the DE2_TV to produce hue, saturation and luminance image of the real time video stream. Figure 11 represents the comparisons of original RGB colour frame with a real time converted video frame of both converter and also with C code converted video frame.

The hardware converted HSL image in Figure 11 did not pose any major difference as compared to the C code converted HSL image. The saturation and luminance image can be considered identical for all hardware and C code conversion image. There is only a minor difference in the hue value located at the center of Figure 11(a) the word ‘K’. It is darker as compared to the Figures 11(e) and (h). This difference is caused by the cyclic properties of the hue. The word ‘K’ is in red colour as depicted in Figure 11(a). Red colour in hue degree is range from 0º to 30º and also 330º to 360º degrees. This large difference of red colour causes the hue value to fluctuate between dark and bright region.
Figure 11. Comparisons of original RGB colour frame with a real time converted video frame of both converters and C code.

4.7 Hardware Resources Consumption
The parallel and pipeline converter hardware resources consumption on the FPGA has been summarized as shown in Table 2. Both of pipeline and parallel architecture converters are only utilize of 3% and 4% from total logic element of the Cyclone II EP2C35F672 FPGA chip respectively.

Table 2. Accuracy of 10 Hz data rate for 1024 RGB samples

| Resources                          | Available | Parallel | Pipeline |
|------------------------------------|-----------|----------|----------|
| Total Logic Element                | 33216     | 1247 (4%)| 950 (3%) |
| Total Combination Functions        | -         | 1247 (4%)| 895 (3%) |
| Dedicated Logic Elements (Register)| -         | 0 (0%)   | 262 (<1%)|
| Total Memory Bits                  | 483840    | 0 (0%)   | 48 (<1%) |
| PLLs                               | 4         | 0        | 0        |
| Total I/O Pins                     | 475       | 50 (11%) | 51 (11%) |
| Embedded Multiplier 9-bits element | 70        | 0        | 3 (4%)   |

5. Conclusion
This work has been successfully developed on both pipeline and parallel hardware architectures for RGB to HSL converter on FPGA. Both of the pipeline and parallel converter architecture functionalities have been verified using Intel-Altera Quartus II functional simulation. Both are further tested in hardware to obtain its accuracy and it has gone through 10Hz slow speed conversion testing to obtain the conversion accuracy. The achievement of accuracy by the parallel and pipeline converters approach are 99.00% and 98.96%, respectively. The HSL output values for these converters are accurate enough to be applied in machine vision system and image processing algorithm. Besides that, HSL output of pipeline architecture is more established than parallel architecture when tested with variable clock frequency from 10MHz to 150MHz. It can be concluded that the parallel converter only capable to handle the data rate up to 50MHz while the pipeline converter is able to handle up to 120MHz. The pipeline architecture is able to speed up twice of parallel architecture. Another performance merit is the parallel converter has consumed 1247 logic elements, 4% of the FPGA resources available while the pipeline converter is only consuming 950 logic elements (3%), 291 logic elements less compared to parallel architecture in the FPGA chip. Therefore pipeline RGB to HSL converter is a better choice with less resources consumption and twice the speed up compared to the parallel converter.

References
[1] Wu, D., and Sun, D.-W. Colour measurements by computer vision for food quality control – A review. Trends in Food Science & Technology 2013; 29(1), 5-20.
[2] Wang, M.-J., Lee, G., and Lin, H.-Y. Extraction of Perceptual Hue Feature Set for Colour Image/Video Segmentation. Advances in Multimedia Information Processing: Springer Berlin Heidelberg 2008; 5353, 875-878.
[3] Aptoula, E., and Lefèvre, S. On the morphological processing of hue. *Image and Vision Computing* 2009; 27(9), 1394-1401.

[4] Tsai, S.H. and Tseng, Y.H. A novel colour detection method based on HSL colour space for robotic soccer competition. *Computers and Mathematics with Applications* 2012; 64(5), 1291-1300.

[5] Hanumantharaju, M. C., Vishalakshi, G. R., Halvi, S., and Satish, S. B. A Novel FPGA Based Reconfigurable Architecture for Image Colour Space Conversion. *Global Trends in Information Systems and Software Applications: Springer Berlin Heidelberg* 2012; 270, 292-301.

[6] Gonzalez, R. C., and Woods, R. E. Digital image processing. *Upper Saddle River, N.J.: Prentice Hall*, 2018

[7] Nnolim, U. A. Design and implementation of novel, fast, pipelined HS2RGB and log-hybrid RGB2HSI colour converter architectures for image enhancement. *Microprocessors and Microsystems* 2015; 39(4–5), 223-236.

[8] Nashat, S., Abdullah, A., Aramvith, S., and Abdullah, M. Z. Support vector machine approach to real-time inspection of biscuits on moving conveyor belt. *Computers and Electronics in Agriculture*, 2011; 75(1), 147-158.

[9] Khamar Basha Shaik, P. Ganesan, V. Kalist, B.S. Satish, and J. Merlin Mary Jenitha, Comparative Study of Skin Colour Detection and Segmentation in HSV and YCbCr Colour Space, *Procedia Computer Science* 2015; 57, 41-48 .

[10] YanJie, Z., and Li, X. A new object detection method in colour image processing. *4th International Congress on Image and Signal Processing (CISP)*, 2011.

[11] Zhong, L., Weihai, C., Yuhua, Z., and Cun, H. Regions of interest extraction based on HSV colour space. *10th. IEEE International Conference on Industrial Informatics (INDIN)*, 2012.

[12] Swenson, R. L. (2000). A real-time high performance universal colour transformation hardware system (BL). (U132261 Ph.D.), *University of Kent at Canterbury (United Kingdom)*, Ann Arbor.

[13] Agoston, M. K. Computer graphics and geometric modelling. Retrieved from http://dx.doi.org/10.1007/b138805, 2015

[14] Pérez, P., Hue, C., Vermaak, J., and Gangnet, M. Colour-Based Probabilistic Tracking. In A. Heyden, G. Sparr, M. Nielsen, & P. Johansen (Eds.), *Computer Vision*: Springer Berlin Heidelberg. 2002; 2350, 661-675

[15] Jung Uk, C., Seung Hun, J., Xuan Dai, P., Dongkyun, K., and Jae Wook, J. IEEE International Conference on FPGA-based real-time visual tracking system using adaptive colour histograms. Paper presented at the *Robotics and Biomimetics*, 2007.

[16] Tsung-Ying, S., Shang-Jeng, T., and Chan, V. HSI colour model based lane-marking detection. *The Intelligent Transportation Systems Conference*, 2006.

[17] Vadiel, A., Mohan, M., Sural, S., and Majumdar, A. K. Segmentation Using Saturation Thresholding and Its Application in Content-Based Retrieval of Images. *Image Analysis and Recognition* 2004; 3211, 33-40.

[18] Na, L., Jiajun, B., and Chun, C. Real-time video object segmentation using HSV space. *International Conference on Image Processing 2002*

[19] Jin-liang, Y., Yi-bo, G., Liang-jun, M., and Yi-Ping, Y. Scene Text Extraction Based on HSL. *International Symposium on Computer Science and Computational Technology* 2008

[20] Abdullah, M. Z., Mohamad-Saleh, J., Fathinul-Syahir, A. S., and Mohd-Azemi, B. M. N. Discrimination and classification of fresh-cut starfruits (Averrhoa carambola L.) using automated machine vision system. *Journal of Food Engineering* 2006; 76(4), 506-523.

[21] Doignon, C., Graebling, P., and de Mathelin, M. Real-time segmentation of surgical instruments inside the abdominal cavity using joint hue saturation colour feature. *Real-Time Imaging* 2005, 11(5–6); 429-442.