Auto Focus Using Adaptive Step Size Search and Zoom Tracking Algorithm

Chia-Hao Chang (張家豪) and Chiou-Shann Fuh (傅楸善)
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan
r92127@csie.ntu.edu.tw and fuh@csie.ntu.edu.tw

ABSTRACT

In this paper, an adaptive step size auto focus search algorithm and a reduced zoom tracking algorithm are proposed.

Many different auto focus search algorithms exist, such as global search, binary search, Fibonacci search, and so on. Global search can ensure the global peak is found correctly, but the search time is too long. The search time of binary or Fibonacci search is shortened, but the lens has to move back and forth frequently, which is prone to have step errors. Our adaptive step size search algorithm performs only one backward movement while the search time is greatly decreased.

Zoom tracking is to adjust a camera's focal length continuously, to keep the in-focus state of an image during zoom operation. A simple table-lookup with interpolation method is commonly used, but the size of the table is important. Our reduced zoom tracking algorithm reduces the table size while still achieving good image quality.

Keywords: Auto focus, Zoom tracking, Adaptive search.

1. ADAPTIVE STEP SIZE SEARCH

1.1 INTRODUCTION

What is focus? If an image is in focus, objects are sharp, clear, finely separated, and details should be highly visible**.** We want DSCs (Digital Still Camera) can find the focus automatically. Auto focus (AF) methods are classified into two categories: active auto focus and passive auto focus methods. Passive auto focus methods are the most widely used auto focus methods. Passive auto focus first calculates every possible lens position's sharpness value also called focus value. Afterward, we search the position which has the maximum focus value, and move lens to the position. Therefore, the passive auto focus methods have two parts: focus value measurements and lens position search algorithms. Figure 1.1 shows the flowchart of passive auto focus methods.

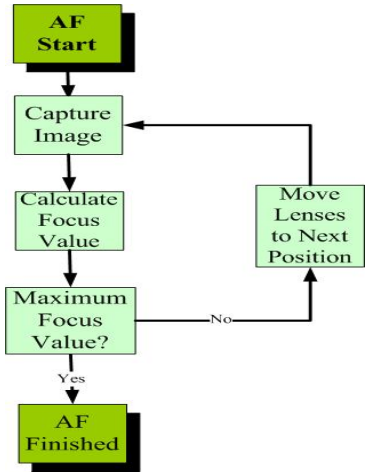


Fig. 1.1. The flowchart of passive AF method.

1.2 FOCUS VALUE MEASUREMENTS

The focus value measurement normally computes the luminance portion of a RGB color filter array pattern, Bayer pattern [1], for instance. However, computing luminance using all three primary RGB colors greatly increases the processing time. Although more contrast accuracy can be obtained by using luminance, in most cases, the use of the green component provides sufficient contrast information for focusing, noting that the green component is the largest contributor to luminance. Thus, we only use the green component to achieve real-time AF operation.

We use IIR (Infinite Input Response) filter as our FV measurement. The advantage of IIR filter is that it is hardware dedicated; the computing speed greatly surpasses software computation. The input data to AF engine are the raw data output by the CCD controller. The AF engine extracts green pixel data from these raw data. The extracted green data are subtracted by 128, and fed to the filter block to calculate the FV, which is the absolute value of the IIR filter output. Value of the green pixels and either the FV or the maximum FV (of

each line) are accumulated in the paxel (a window of green data), and are stored in SDRAM [16]. We separate the image into 12x8 blocks (paxels.) The DM 320 processor generates CCD raw data of size 3004 x 270 in its preview mode. The size of our AF statistics windows is 1920 x 240; however, we omit the border windows to correctly measure the subject. The exact windows are shown in Figure 1.2.

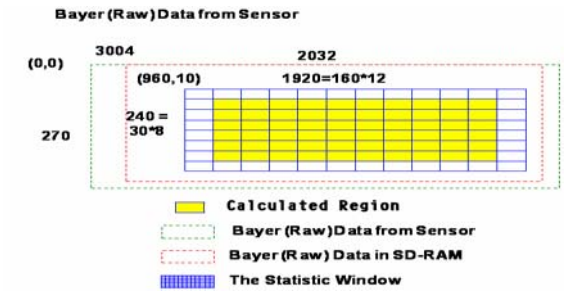


Fig. 1.2. The AF statistics windows

1.3. ADAPTIVE STEP SIZE SEARCH

In this section, our adaptive step size search (AS) will be introduced. We categorize the lens movement step into four situations: *Initial*, *Coarse*, *Mid*, and *Fine* states. Section 1.3.1 will introduce the step states; Section 1.3.2 will explain the decision rules of the states.

1.3.1 Step state

We categorize the lens movement step into *Initial*, *Coarse*, *Mid*, and *Fine* states. The *Initial* state is the initialization of our algorithm. In this state, we compute the FV average of the first five steps as a criterion for later use and initialize all variables.

In the *Coarse* state, we consider the FVs in this state are stable, and the FV curve is flat. The existence of global peak is almost impossible in this state. Therefore, we can skip most of them while searching; every 8 to 12 steps are examined.

In the *Mid* state, the FV curve might have the local peak and the FVs change more. These characteristics make us more careful when searching in this state. The local peak might be the global peak. We cannot just skip the entire local peak, like in the *Coarse* state. However, the possibility to contain a global peak is still small. If we search every step in this state, the total search time will be too long. Therefore, instead of searching step by step in this state, we search every 3 to 5 steps to both save time and not lose the information in this state.

Fine state represents high possibility of containing global peak. In this state, FVs change violently; even two nearby lens positions' FVs are obviously different. The FV curve goes up or down sharply. Therefore, once we consider the search enters *Fine* state, we stop skipping steps, which we always do in the previous two states. We examine every step in this state to ensure the global peak is finally found, and the FV curve shows the global peak completely.

With these states, our adaptive step sizes are defined. We next decide searching states. How to decide the FV curve is in *Coarse*, *Mid*, or *Fine* state? Our adaptive step size searching defines it in Section1.3.2.

1.3.2 Decision criteria

In this section, we will explain how we decide the states in detail. While the auto focus starts, our first criterion is to judge whether the search position is the first five steps or not. If yes, then enter the *Initial* state. In the *Initial* state, we sum the first five steps' FVs. Afterwards, we first calculate FV_{avg}, the average FV of the first five steps.

After the *Initial* state, we start calculating FV_{dif}, the difference between the FV_{cur} and FV_{pre}, the current and previous frames' FVs respectively, in every frame to be examined. We also keep updating the FV_{max}, the maximum FV so far, in every frame. After calculating the FV_{avg}, FV_{dif}, and FV_{max}, we finish preparing our adaptive step size search. We now can judge whether a frame is in *Coarse*, *Mid*, or *Fine* state by FV_{avg}, FV_{dif}, and FV_{max}. Note that once a frame is set to a state, whether is *Initial*, *Coarse*, *Mid*, or *Fine* state, this frame is finished and we start with next frame.

Here we introduce two more variables, *DownCount* and *MidCount*. Every time when FV_{dif} is smaller than zero in *Fine* state, *DownCount* should add one. If the state is not *Fine*, then the *DownCount* should be set to zero every time. *MidCount* should add one while the FV_{cur} is smaller than FV_{max} in the *Mid* state. The usage of these two variables will be introduced later in this section.

After the above initialization, the remaining frames are categorized into *Coarse*, *Mid*, or *Fine* state. The second criterion considers an incoming frame's FV, FV_{cur}. If FV_{cur} is smaller than a threshold, T_c, the *Coarse* state threshold, times FV_{avg}, or FV_{cur} is smaller than 0.7* FV_{max}, we take this frame's FV as smaller enough to be in *Coarse* state, and *DownCount* is set to zero. If not, we consider the next criterion. This criterion normally takes effect at the front part of FV curve.

The Criterion 3 is to decide *Mid* state. If FV_{cur} is larger than T_c*FV_{avg}, smaller than T_m, the

Mid state threshold, times FV_{avg}, and FV_{cur} is bigger than FV_{max}, the state is set to *Mid*, and *DownCount* is set to zero. This criterion also normally takes effect at the front part of FV curve.

Another *Coarse* state's criterion, Criterion 4, is introduced in this paragraph. If the FV_{cur} is smaller than FV_{max}, *MidCount* larger than 3, *DownCount* is not equal to 3, and the current state is not *Fine*, the state is set to *Coarse*, *DownCount* and *MidCount* is set to zero. If *DownCount* equals 3 or more, we consider it as the downhill peak, the actual peak is found, and we change the state to *Mid*. If *MidCount* is larger than 3, we consider it as the chance of containing a peak gets smaller. Therefore, we change to *Coarse* state. The difference between Criteria 2 and 4 is that criterion 4 normally takes effects at the end part of FV curve.

If the state is not decided, then we use Criterion 5, FV_{dif}, to judge. The flowchart of the upper part of our proposed adaptive step size algorithm is shown in Figure. 1.3(a).

If FV_{dif} is larger than or equal to zero, than we go to *A* in Fig. 1.3(b); otherwise, we go to *B* in Fig. 1.3(b). We first consider Situation *A*. Criterion 6 considers that if FV_{dif} is larger than T_{dif}, the difference threshold, times FV_{pre}, and FV_{cur} is larger than FV_{max}. If yes, then we set state to *Fine*, and *DownCount* is set to zero. Otherwise, we consider next criterion. This criterion normally takes effect during the peak in FV curve.

Criterion 7 examines whether FV_{dif} is smaller than T_{dif}* FV_{pre}, and the previous state. If yes, then the state is set to *Mid*, *DownCount* is set to zero and Criterion 9 checks if FV_{cur} is smaller than FV_{max}, if yes, then *MidCount* should add one.

Criterion 8 examines whether FV_{cur} is smaller than 0.9* FV_{max}, if yes, then the action is just like Criterion 7. If none of them take action, then the state remains and moves to next frame.

Now we consider Situation *B*. Criterion 10 examines if the state is *Fine*. If it is, then the state remains, but the *DownCount* adds one because of the FV_{dif} smaller than zero.

If the *DownCount* equals three, or FV_{cur} smaller than 0.9* FV_{max}, the last criterion, Criterion 11, takes effect. *DownCount* is reset to zero and the state is set to *Mid*, which brings in Criterion 9.

After the state is decided, the lens moves according to the state, and the FV_{pre}, and FV_{max} are updated. If the search range is finished, we track the FV buffer and find the maximum FV and its related step position. Auto focus is finished when the lens moves to the right position, which is the peak. Fig. 1.3(b) shows the flowchart of this part.

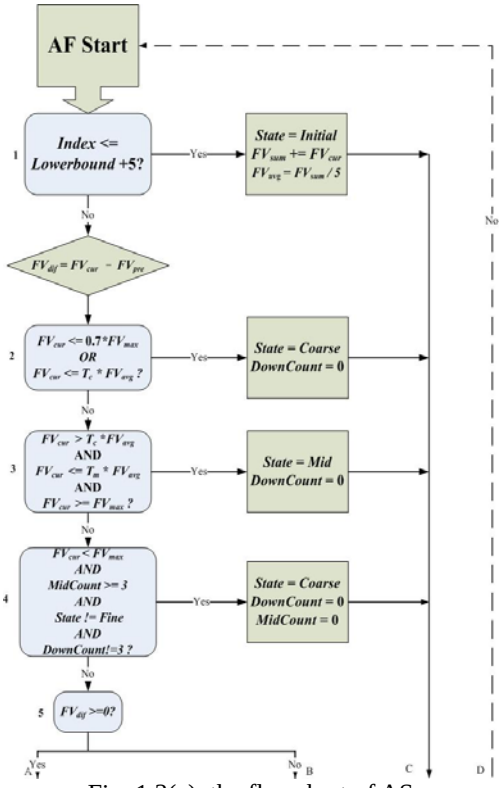


Fig. 1.3(a). the flowchart of AS.

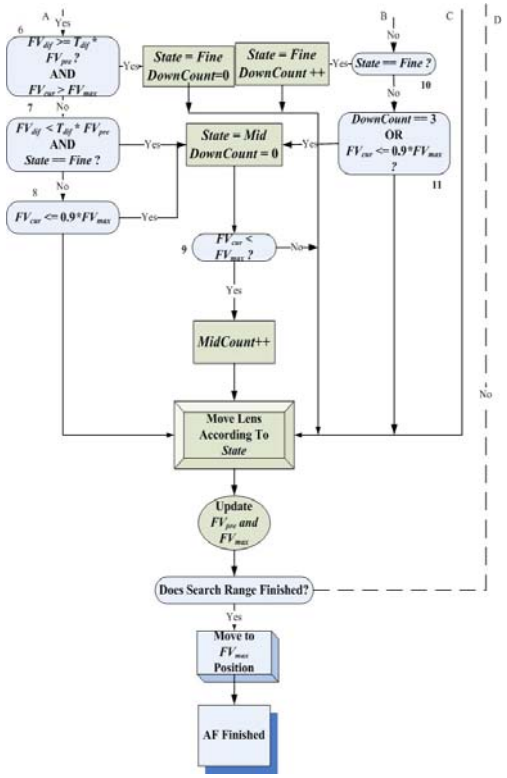


Fig. 1.3(b). the flowchart of AS.

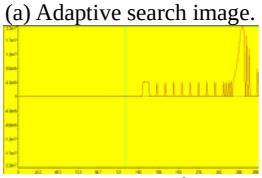
1.4 EXPERIMENT RESULTS

In this section, we will show the results of our adaptive step size search (AS) algorithm. The results will be compared with global search in the actual image, FV curve, total search iterations, and total steps. The search time is directly related to total search iterations. More iterations mean longer search time. Another performance issue is the mechanical backlash problem. Many stepping motors, including ours, suffer from this problem. For our stepping motor, each backward movement requires 32 steps moves, 16 backward and 16 forward steps [14]. Therefore, searches which require moving lens back and forth frequently, like binary search or Fibonacci search, suffer from this problem severely. More steps means the more power consumption, since every movement of steps needs power support. Both our algorithm and global search use IIR filter with same coefficients as the FV measurements, and capture the same windows, same parts of an image.

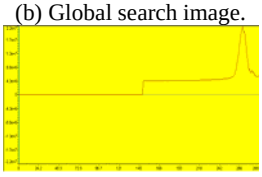
1.4.1 Experiment 1

(a) Adaptive search image.

(b) Global search image.



(c) FV curve of AS
Total search iterations: 37
Total move steps: 69

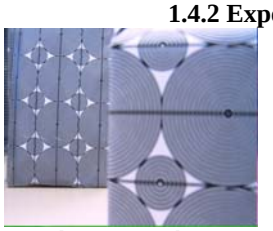


(d) FV curve of GS
Total search iterations: 140
Total move steps: 172

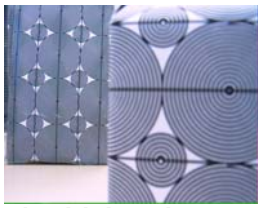
Fig. 1.4. The result of Image 1.

This is the most common image. The subject is at center with lots of high frequency components. We can see that Figure 1.4(a) and (b) look very similar. Figure 1.4(c) shows the FV curve of AS. Compare it with Figure 1.4(d), we can understand that the AS algorithm can find the global peak while saving many steps. Moreover, the total search iterations and total movement steps are both very small. Only one backward movement of AS and GS is needed. Because both searches store FVs in the buffer, search the maximum FV in the buffer, and the lens moves to the related position.

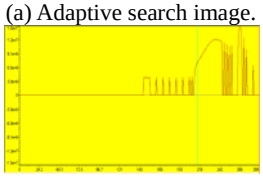
1.4.2 Experiment 2



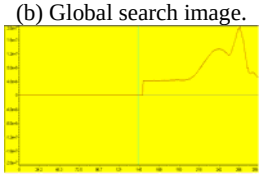
(a) Adaptive search image.



(b) Global search image.



(c) FV curve of adaptive search.
Total search iterations: 58
Total move steps: 90



(d) FV curve of global search.
Total search iterations: 140
Total move steps: 172

Fig. 1.5. The result of Image 2.

This image has two main peaks. The smaller peak is at front, while the larger one at end. Our AS algorithm gets the same results with GS: both focuses at the post subject in (a); (c) shows FV curve of AS. We can observe that it first finds the previous peak, and goes downhill, *Mid* state, and switches to *Fine* state to find another peak. From this image we can learn that our AS algorithm can find the real peak even there exists a local peak.

1.4.3 Summary

To summarize, the average total search iterations of AS are 50 iterations and the average total move steps are 82 steps. The AS saves 90 iterations and almost halves the total moving steps by average comparing with GS.

The adaptive step size auto focus search algorithm is a passive auto focus algorithm in DSC to find the global peak of FV functions. The performance of our algorithm has been shown as an efficient algorithm in terms of searching time and power consumption.

1.5 FUTURE WORK

In our algorithm, the search states are decided by some thresholds, such as T_c, T_m, and T_{dif}. However, the values of these thresholds are hard to define. Thresholds in the bright light situation and low light situation are totally different. Integration of luminance information might improve the performance. Different luminance values with different threshold values can avoid the wrong decision of the states.

2. REDUCED ZOOM TRACKING

2.1 INTRODUCTION

Zoom tracking adjusts a camera's focal length continuously, to keep the in-focus state of an image during zoom operation. Zoom tracking is important especially in video cameras or camcorders because we want the subjects always in focus even when we are zooming in or out. In DSC, zoom tracking is less important than in video cameras or camcorders, but we still want the subjects always in focus in order to shorten the lens moving range while auto focusing. The zoom tracking technique can be implemented using a simple curve traced table-lookup method [8]. We can store several zoom position curves with respect to the in-focus lens position, and move the focus lens position by looking up the position table while moving the zoom lens position. Figure 2.1 shows the lens position curve. However, the table-lookup method needs a large system memory, which is often limited in portable devices. Another problem of zoom tracking is that the lens position curve selection gets harder while the zoom lens moves toward the telephoto angle. The de-focusing gradually increases as the zoom lens moves toward the telephoto end. We will discuss several existing methods in Section 2.2, and propose our zoom tracking algorithm in Section 2.3.

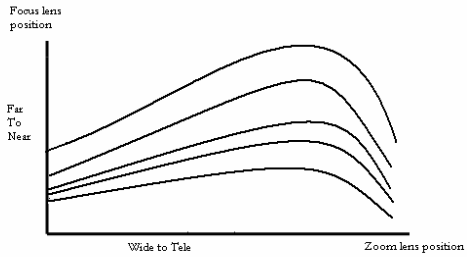


Fig. 2.1. An example of zoom tracking lens position curve.

2.2 RELATED WORKS

This algorithm uses the curve interpolation and estimation techniques. Each curve is divided into the linear and non-linear regions as shown in Figure 2.2. In the linear region, the left and right end points are stored in the memory and the rest focus positions are calculated from the two points using the linear interpolation method. In the nonlinear region, the focus position at each zoom position is obtained from the stored curve data.

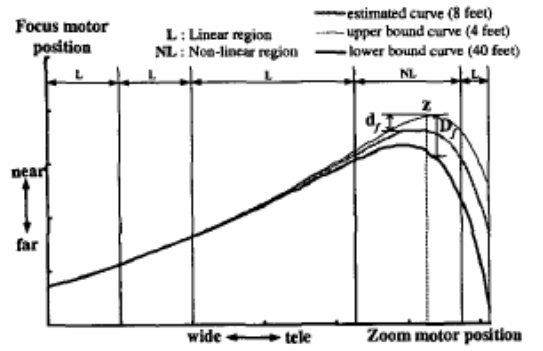


Figure 2.2. The adaptive zoom tracking [12].

Curves between the upper and lower bound are estimated as in Equation (2.1):

$$F(k) = F_1(k) - R * D_f \qquad (2.1)$$

where $F(k)$ and $F_1(k)$ are the focus position of the estimated and upper bound curves at zoom position k, respectively; R is the curve estimation factor of d_f/D_f , where D_f is the difference between focus position of the upper and lower bound curves at the zoom position k, and d_f is the difference of the focus position between the upper bound and the estimation curve at the same position.

The algorithm initially traces the upper bound curve since the difference between the focus positions of each curve is very small in the linear region; in non-linear region, the curve estimation factor is calculated. Finally, zoom curve interpolation and estimation are performed using the curve estimation method.

2.3 REDUCED ZOOM TRACKING

In this section, we will introduce the reduced zoom tracking (RZT). Our algorithm also uses adaptive methods, like Section 2.2. However, the adaptive zoom tracking in Section 2.2 only traces the upper bound curve, where the loss of steps will get larger while the curve is closer to the lower bound. We propose another partition method to improve this problem.

We first cut the zoom step range into nine zoom positions, zoom 1 to zoom 9. For every zoom positions, we measure the focus step of nine different subject distances to construct zoom table. Therefore, we have 9* 8= 72 data shown in Figure 2.3.

Our goal is to reduce the data while still performing good zoom tracking image quality. We divide the data curve into *Linear Region 1*, *Linear Region 2*, and *Nonlinear Region*, which means that we separate the data into three tables. However, we do not keep all the data; we only reserve the

complete data in the *Nonlinear Region*. The implementation detail will be explained in Sections 2.3.1 and 2.3.2.

2.3.1 Linear Region

In *Linear Region 1*, we extract the first four zoom positions' focus step from the original data. We will have 4*9 =36 data, but we do not store all of them. Instead, we separate nine subject positions into three groups, the lower curve, the middle curve, and the upper curve. The lower curve consists of the first two subject positions, which is 0.4m and 0.5m, for nine zoom positions, and the values are gotten from averaging the values of two close subject positions, and both subject positions are set to the same value. The middle curve contains the middle four subject positions, 0.6m, 0.8m, 1m, and 1.5m. The value is set to the average value. The upper curve contains the last two subject positions, 2.5m and infinity, and set to the average value. In this way, the original 36 data are reduced to 3 (lower, middle, and upper curve) * 4 (zoom positions) =12 data, where 20 data are saved. The three curves are shown in the *Linear Region 1* in Figure 2.4.

Linear Region 2 uses the same concept to divide the data for the middle two zoom positions, which are 2*9=18 data. In this region, data are more dispersed. Therefore, we add one curve, bottom curve, for accuracy. Bottom curve conserves the original data of 0.4m. Lower curve sets the value by averaging 0.5m and 0.6m's focus step. Middle curve contains 0.8m, 1.0m, and 1.5m. Upper curve uses the average value of 2.5m and infinity. Therefore, only 4 (value)*2 (zoom positions) =8 data will be stored in this region, which are 10 data saved. The four curves are shown in the *Linear Region 2* in Figure 2.4.

Once the data of *Linear Region 1* and *Linear Region 2* are stored, we can find our zoom tracking focus steps by directly looking up the table. The estimated curve's focus step is obtained by comparing the previous focus step to tables, finding its related focus step index, and getting next zoom position's focus step directly from the table.

2.3.2 Nonlinear Region

In the nonlinear region, all curves are almost separated, so all data of the last three zoom positions are stored. The estimated curve's focus step is set by curve interpolation method in Equation (2.2).

$$Estimated = LowerBound + D_f \frac{d_s}{D_s} \quad (2.2)$$

where D_s is the difference between the focus positions of the upper and the lower curves; d_s is the difference between the focus positions of the estimated and the lower curves at the zoom start point; D_f is the difference between the focus positions of the upper bound and the lower curves at current zoom point.

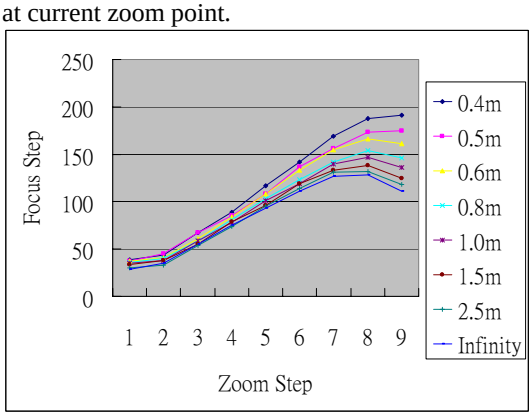


Fig. 2.3. The zoom tracking curve of original data.

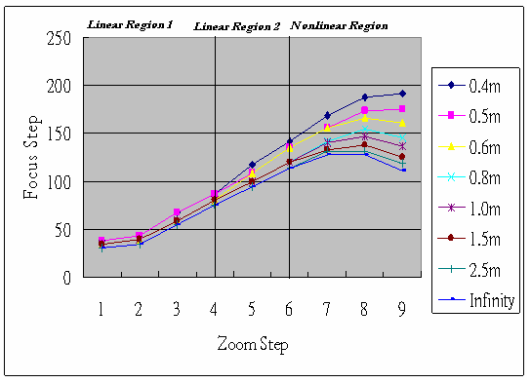


Fig. 2.4. The zoom tracking curve of reduced data.

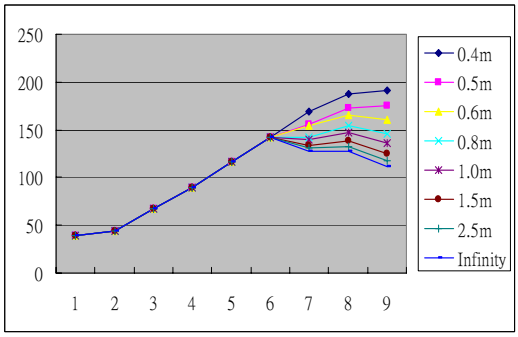


Fig. 2.5. The zoom tracking curve of UZT.

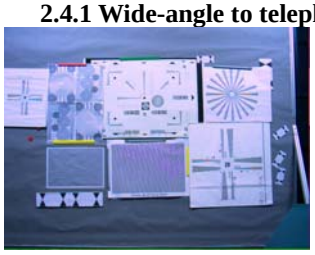
2.3.3 Summary

Combining Sections 3.2.1 and 3.2.2, we can obtain that our stored data are 12+8 (linear region) + 9*3 (nonlinear region) = 47 data and the original data are 72 data. Therefore, we reduce the data by 47/72=65%, almost 1/3 data are saved.

2.4 EXPERIMENT RESULTS

In this section, we will show our experimental results of our reduced zoom tracking algorithm, and compare it with adaptive zoom tracking which just traces the upper bound in the linear region (we call it UZT). Its zoom tracking curve will be shown in Figure 2.5. We will show our results in visual comparison, and focus step error comparing with the focus step found by global search. Section 2.4.1 explains the images from wide-angle to tele-photo, while Section 2.4.2 discusses the images from tele-photo to wide-angle. Section 2.4.3 is the summary.

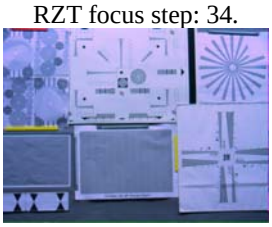
2.4.1 Wide-angle to telephoto



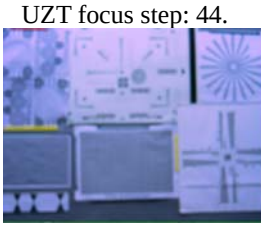
(a) Original image with no zooming.
GS focus step: 26.

(b) RZT zoom 1.
GS focus step: 32.
RZT focus step: 34.

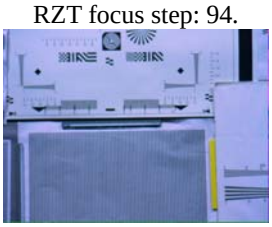
(c) UZT zoom 1.
GS focus step: 32.
UZT focus step: 44.



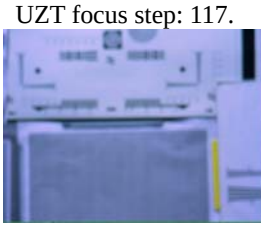
(d) RZT zoom 4.
GS focus step: 90.
RZT focus step: 94.



(e) UZT zoom 4.
GS focus step: 90.
UZT focus step: 117.



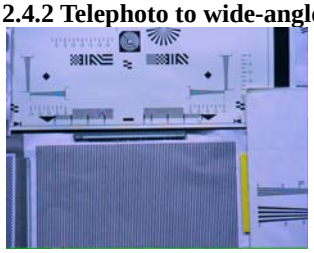
(f) RZT zoom 8.
GS focus step: 108.
RZT focus step: 119.



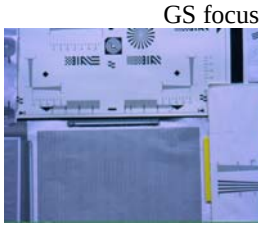
(g) UZT zoom 8.
GS focus step: 108.
UZT focus step: 147.

Fig. 2.6. The results from W to T.

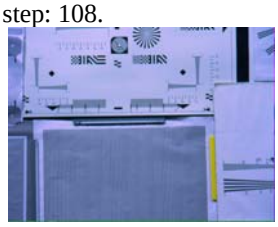
2.4.2 Telephoto to wide-angle



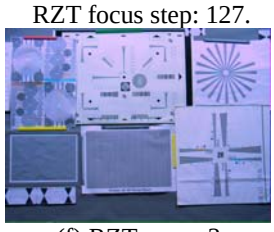
(a) Original image with global search zoom 8.
GS focus step: 108.



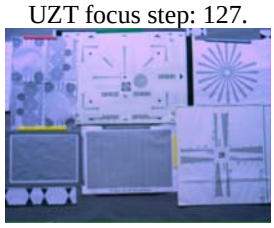
(b) RZT zoom 7.
GS focus step: 125.
RZT focus step: 127.



(j) UZT zoom 7.
GS focus step: 125.
UZT focus step: 127.



(f) RZT zoom 3.
GS focus step: 72.
RZT focus step: 75.



(n) UZT zoom 3.
GS focus step: 72.
UZT focus step: 89.

(i) RZT zoom 0.
GS focus step: 26.
RZT focus step: 30.

(q) UZT zoom 0.
GS focus step: 26.
UZT focus step: 39.

Fig. 2.7. The results from T to W.

2.4.3 Summary

The UZT algorithm stores 1(focus position) * 6(zoom positions) + 8(focus positions)* 3(zoom positions) = 30 data. Although the UZT use 30/72 = 42% data of original, better than our 65%, the performance of RZT from both wide-angle to tele-photo or tele-photo to wide-angle, are better than UZT, especially in the linear region, which achieves our goal. Our RZT algorithm is shown to be more accurate than UZT while still shrinking the data size in both lens moving directions. We compare the average steps loss with two more methods, UZT with middle curve, and UZT with lower curve, which traces middle or lower curve, respectively. The comparisons in detail are shown

in Fig. 2.8 and Fig. 2.9.

Object is far (> 1m)	RZT	UZT	UZT with middle curve	UZT with lower curve
Data usage with original data	65%	42%	42%	42%
Average steps loss from W to T.	5.125	24	21.625	12.75
Average steps loss from T to W.	3.125	15.25	12.125	2.75

Figure 2.8. The comparison results (a).

Object is near (<= 1m)	RZT	UZT	UZT with middle curve	UZT with lower curve
Data usage with original data	65%	42%	42%	42%
Average steps loss from W to T	3.375	10.125	15.25	19.875
Average steps loss from T to W.	2.375	2.75	9.375	14.875

Fig. 2.9. The comparison results (b).

2.5 FUTURE WORK

In the direction from the wide-angle to tele-photo, we can focus well in the linear region, but in non-linear region, the image quality can be improved. In another direction, tele-photo to wide-angle, can the interpolation method be improved? These are some issues that can be discussed.

ACKNOWLEDGEMENT

This research was supported by the National Science Council of Taiwan, R.O.C., under Grants NSC 94-2213-E-002-032 and NSC 93-2213-E-002-073, by theEeRise Corporation, EeVision Corporation, Machvision, Tekom Technologies, IAC, ATM Electronic, Primax Electronics, Scanace, Lite-on and Liteonit.

REFERENCE

[1] B. E. Bayer, “Color Imaging Array,” US Patent No. 3971065.

[2] G. S. Beveridge and R. S. Schechter, *Optimization: Theory and Practice*, pp. 180-193, McGraw-Hill, New York, 1970.

[3] V. Bockaert, “Focal Length,” http://www.dpreview.com/learn/?/Glossary/Optical/Focal_Length_01.htm, 2003.

[4] Bores Signal Processing, “Introduction to DSP,” http://www.bores.com/courses/intro/iir/5_eq.htm, 2004.

[5] Canon, “Calculate Depth of Field,” http://www.canon.com/bctv/calculator/calculator2.html, 2005.

[6] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd Ed., pp. 128-132, Prentice Hall, New Jersey, 2002.

[7] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, Vol. I, pp. 337-352, Addison-Wesley, Reading, MA, 1992.

[8] P. Hoad and J. Illingworth, “Automatic Control of Camera Pan, Zoom and Focus for Improving Object Recognition,” *Proceedings of IEE International Conference on Image Processing and Its Applications*, Edinburgh, no. 410, pp. 291-295, 1995.

[9] Iowegian International, “Infinite Impulse Response Filter FAQ,” http://www.dspguru.com/info/faqs/ iirfaq.htm, 2004.

[10] N. Kehtarnavaz and H. J. Oh, “Development and Real-Time Implementation of a Rule-Based Auto-Focus Algorithm**,”** *Journal of Real-Time Imaging*, Vol. 9, Issue 3, pp. 197-203, 2003.

[11] Y. Kim, J. S. Lee, A. Morales and S. J. Ko, "A Video Camera System with Enhanced Zoom Tracking and Auto White Balance," *IEEE Transactions on Consumer Electronics*, Vol. 48, No. 3, pp. 428 -434, 2002.

[12] J. S. Lee., S. J. Ko, Y. Kim, and A. Morales, “A Video Camera System with Adaptive Zoom Tracking,” *Proceedings of International Conference on Consumer Electronics*, Los Angeles, California, pp. 56-57, 2002.

[13] A. Macbeth, “Camera Basics,” http://www.Andrewmacbeth.com/articles/camera_basics.htm, 2005.

[14] Ricoh, *4T16-3X-3.3M Lens Module Sample Specification Sheet*, 2002.

[15] A. R. Sampson, “Scanning Electron Microscopy,” http://www.sem.com/analytic/sem.htm, 1996.

[16] Texas Instruments, “TMS320DM320 Image Peripherals Vol-1b,” Technical Reference Manual, Ver. 1.4, 2005.