Semi-automated binary segmentation with 3D MRFs

1st Shida Sheng
Trinity College
Electronic Information Engineering Of MSC
Dublin, Ireland
shengs@tcd.ie

***Abstract*—In this paper, it present a semi-automated method for doing binary segmentation on continuous frame. Based on the traditional segmentation method of Maximum Likelihood algorithm and the Markov random field algorithm, this method implement a optimization algorithm by using the motion vectors information from the previous frame to do the image segmentation.It not only considers the advantages of spatial factors for graphic segmentation, but also takes into account the impact of motion vectors on continuous frames on segmentation,it takes into account the effects of temporal factors.**

***Index Terms*—Markov random field algorithm,Motion vectors,Temporal factors, Graphic segmentation.**

I. Introduction

Compositing is a process of combining different visual elements together into a single images.This technique is commonly used in the film industry for the production of special effects in movies, which include scene transformations.In the process of film production, there may be millions of frames that need to produce special effects on the film. If the graphic segmentation work is rely on man-made, it is very time-consuming and the accuracy of graphic segmentation is not guaranteed.On video segmentation, each frame is not a still image,the motion in each frame must have influence on the segmentation result. There have some key references to Bayesian Matting [1], like the Bayesian Approach to Digital Matting published on 2001 and Markov Random Fields for Vision and Image Processing. Edited by A.Blake,P.Kohli and C.Rother, MIT Press,2011 [2]. The most difference between these paper to the method we implemented is using the spatial information in the continuous frame,which is the motion vectors.

II. Colour segmentation

The colour segmentation is a technique in imaging processing area. By given a image, how to separate a specific area from the background image? The image can be considered as a big matrix, what we do is to find out the difference between the specific area array and the background area array. It has many way to figure out this problems, but all the solution is to discover $\alpha(x)$ [3] .The color keying can be described as the equation below.

$$Y = F \times \alpha + B \times (1 - \alpha) \tag{1}$$

Trinity College.

where x is the position of the pixel with value I(x), Let $\alpha(x)$ be the binary matte value at that site. Foreground is indicated by $\alpha(x) = 1$.

III. Bayesian Algorithm

A. Maximum Likelihood Algorithm

This approach [1] [4]is to use the background image of known color(typically green) to make some certain assumptions.The background color can be assumed that it follow the Gaussian distribution.This step can help to estimate the foreground colors along the boundary.If the color is given,the opacity value is determined.the problem is to estimate b(x) a binary segmentation mask which is b(x) = 0 in the background and b(x) = 1 in the foreground. First, Measure Gaussian parameters for background(some region) in the image. Calculate the mean and variance of the image. Second, Set a threshold E_t for controlling the whether a site is background or not. The threshold control the quality of the graphic segmentation. Third, At every pixel site, Calculate the equation below, Where the r,g,b stand for the image channels, E(x) means the probability of the Bayesian mask(likelihood energy).

$$E(x) = \frac{(B_r - \bar{B}_r)^2}{2\sigma_r^2} + \frac{(B_g - \bar{B}_g)^2}{2\sigma_g^2} + \frac{(B_b - \bar{B}_b)^2}{2\sigma_b^2} \tag{2}$$

$$E(*) < E_t, \alpha = 0, or, \alpha = 1 \tag{3}$$

IV. Markov random field

Markov random field [5] [2] is established in signal processing by Jogn Woods and it is popularly used in imaging and speech processing, It leads to well understood optimisation schemes and behaviour. It usually use the Bayesian inference as the prior probability. The key idea of Markov random field is to find out things about the current site, it just focus on the difference between the site and its neighbourhood. It can be described as the equation below.

$$p(b(x)|B) = p(b(x)|B(x\epsilon\mathbb{N})) \tag{4}$$

A. 2D Markov random field

To implement the 2D Markov random field [6], the first step is to choose a specific region to calculate the Gaussian parameters (mean and variance) of this region. After getting the Gaussian parameter of the region, it will be used as the prior probability and compare the foreground site with it neighbourhood to decide whether the site is belong to foreground or background. The key improvement is to calculate the spatial energy. This process of the algorithm [7] can be described as the equation below.

$$P(a(x)|N_a(x)) = \frac{1}{Z} exp - \Delta[\sum_{k=1}^{n} \lambda_k |a(x) \neq a(x+q_k)] \quad (5)$$

- where n is the total pixel number,q_k is the the location of the neighbourhood pixel

B. Optimal solutions

It has only the conditional probability of one alpha value at one site GIVEN the current state of all the other alphas. In order to maximise the joint probability for all the variables(alpha) at all site, the Iterated Conditional Modes(ICM) [8]is a good method to solve this problem. It pick the best value for alpha at one site then with that new value, visit the next site. Then iterate. It chooses the variable which maximises the local conditional density, and then doing the same at every site until the converges comes to the minimum. After we use the Iterated Conditional Modes, the algorithm can be described as Maximum a-Posteriori (MAP) Estimate, the difference to the Maximum Likelihood Algorithm is the probability of background and foreground is consist of the equation(3), the probability equation can described as below.

- For iteration 1 : N.

$$E_l = E(x) = \frac{(B_r - \bar{B}_r)^2}{2\sigma_r^2} + \frac{(B_g - \bar{B}_g)^2}{2\sigma_g^2} + \frac{(B_b - \bar{B}_b)^2}{2\sigma_b^2} \quad (6)$$

$$E_s(0) = \sum_{k=1}^{n} V(0, a(x+q_k)) \quad (7)$$

$$E_s(1) = \sum_{k=1}^{n} V(1, a(x+q_k)) \quad (8)$$

$$E(0) = E_l + E_s(0) and E(1) = E_t + E_s(1) \quad (9)$$

If

$$E(0) < E(1), \alpha = 0, or, \alpha = 1. \quad (10)$$

Where N is the iteration times, n is the number of the neighbourhood pixels, E_t is a constant threshold. The E_s is the spatial energy for the background and foreground.

C. Iteration conditional modes

When we do the iteration conditional modes [8], the result goes be better and better, but when the iteration times comes to a certain value, the result performance will not have any improvement.

D. 3D Markov random field

3D Markov random field means three dimension markov random field, it considers the temporal factors more than the 2D markov random field. In video processing, there has a very small difference between frame to frame, the most difference is caused by the motion in each frame. If there have three continuous frame, n-1,n,n+1,among these three frame, we do the motion estimation to get the motion vector from frame to frame. This motion vectors stands for the the object moving trajectories. By using the motion vectors, it can generate the motion compensation image. For example, we can use the motion vector from n-1 to n to generate the prediction image of n. The motion compensation image is generated by the n-1 frame and motion vector [9], it can be described as the equation below.

$$I_n = I_{n-1}(x + d_{n,n-1}) \quad (11)$$

- where the I(n) is the n frame, I.(n-1) is the n - 1 frame, d is the motion vector from n - 1 to n.

Comparing with 2D Markov random field, the 3D Markov field use the motion vectors to optimize the result, it has a better accuracy than the 2D Markov field. The image below can shown how it works.

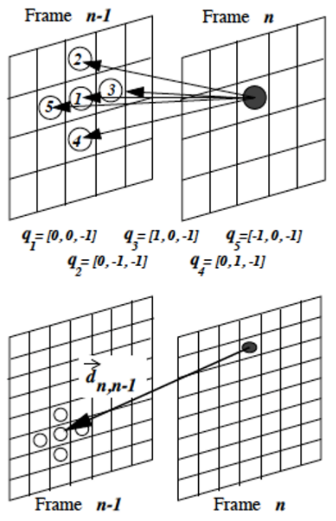


Fig. 1: 3D Markov random field.

- The Fig.1 [10] clearly shows the backward vectors from the Frame n to Frame n-1.

V. My work

A. Implement the Bayesian approach

The Bayesian approach [1] is the first step in doing this assignment. Before the Bayesian approach, we can convert the image into another colourspace. The YUV colorspace is more related to the perception of human,Y stands for the luma component(the brightness) and U and V are the chrominance (color) components. The conversion equation is shown below.

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} * \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \tag{12}$$

- First,choosing a specific region and calculating the Gaussian parameter(mean and variance) from the background image.Because the image usually have three channels, we will get three pairs of mean and variance respectively. The equation of mean and variance is shown below.

$$\bar{X} = \frac{\sum_{i=1}^{N} X_i}{N} \tag{13}$$

$$\sigma^2 = \frac{\sum (x_i - \bar{x})^2}{N} \tag{14}$$

- Second, putting all the mean and variance into the equation(1) to calculate the probability of the Bayesian mask. The mask is shown on the last page.(likelihood energy)
- The last step of the Bayesian approach is to set a threshold to control whether the site is belong to foreground or background. A properly value can lead a perfect outcome. I use as my threshold value.

B. Implement 2D MRF

The 2D Markov random field approach is based on the Bayesian approach to compare the specific site with its neighbourhood. The steps is shown below.

- After getting the result from the Bayesian approach, we will take the result to calculate the spatial energy by using the equation(5)(7)(8).
- Using the background spatial energy,foreground spatial energy and the likelihood energy to discover the $\alpha(x)$, this process is followed the equation(7)(8)(9)(10).

C. Implement 3D MRF

- The 3D Markov random field approach has the advantage of temporal factor and it is more related to video processing. The key in 3D Markov random field approach is to do the motion estimation and motion compensation. The motion estimation generate the motion vector from frame to frame, by using these motion vectors,we can generate the prediction image. In 3D Markov random field, we will calculate the likelihood energy and spatial energy on the prediction image. This process can be described as the equation below.

D. Implement Iterated conditional modes

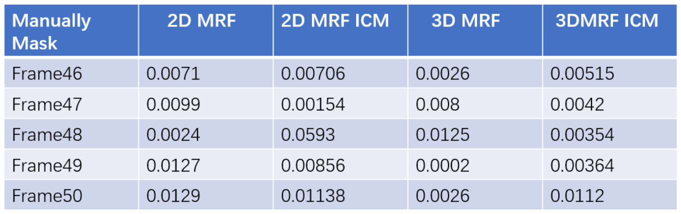
- When we obtain the result from the Markov random field approach, the binary matte still have some problems in its edge area. In order to optimise the result, we can use the iterated conditional mode to obtain the better result. The result from the first time iteration will pass to the second time iteration and as the input of the second time iteration, we will keep doing this until the result can get the best $\alpha(x)$

E. Performance

In order to evaluate the performance among these three approaches,The solution is manually segment the 5 frames in the sequence and then measure how close the estimated mask is to that. By using the Matlab, I compare with the difference between the different approach and the manually segment mask. I write a Matlab code to search every pixel in the image and record the number of ”white” pixel in different appraoch to evaluate the performance between different approaches.

Manually segment Mask	**The performance**		
	Bayesian Approach	***2D MRF***	***3D MRF***
Frame0046	0.0287	0.0071	0.0026
Frame0047	0.0393	0.0099	0.008
Frame0048	0.0440	0.0024	0.0125
Frame0049	0.0320	0.0127	0.0002
Frame0050	0.0334	0.0129	0.0026

The table shows the performance among Bayesian approach, 2DMRF and 3D MRF. The data in the table means how different approach close to the manually segment Mask.(The smaller the number, the closer to the Manually mask)
Now we can also observe the performance of the 2DMRF ICM and 3D MRF ICM.



Manually Mask	2D MRF	2D MRF ICM	3D MRF	3DMRF ICM
Frame46	0.0071	0.00706	0.0026	0.00515
Frame47	0.0099	0.00154	0.008	0.0042
Frame48	0.0024	0.0593	0.0125	0.00354
Frame49	0.0127	0.00856	0.0002	0.00364
Frame50	0.0129	0.01138	0.0026	0.0112

Fig. 2: ICM Performance.

After doing the Iterated conditional modes, the result have some improvment on the performance, But it is still limited by the original result. Although the 3D MRF result is better than the 2D MRF result, the cost of 3D MRF is much bigger than the 2D MRF. When nuke implements the 2D MRF, it takes less time than 3D MRF. The calculation in 3D MRF is twice as the calculation in 2D MRF. Theoretically,it should take more than twice time as the 2D MRF approach. In order to observe the difference between different approach result, i use a nuke test script to show the difference between different approach result.The image will illustrate in the last page.

CONCLUSIONS

In all these approaches to do the Color segmentation, we are trying to find out the keying α and using the equation(1) to do the compositing work. All these approaches is based on finding the parameter of the background image. It means if the background color is not a single specific color, these approaches is useless. The image we used is a lady who is wearing a green dress and also have some green tatto on her face. It must have a influence on the matting performance. If the background color is using a single color(such as green), the person should not wear any similar color(green) clothes.

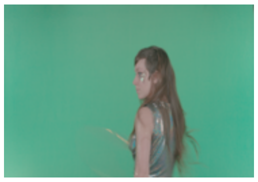


Fig. 3: One frame of Hula

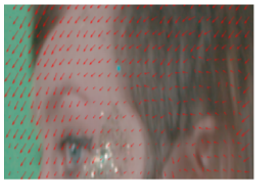


Fig. 4: the motion trajectories

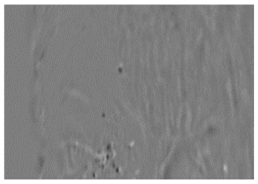


Fig. 5: Motion Compensation frame difference

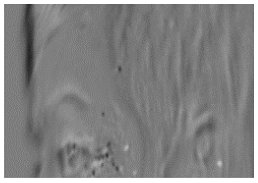


Fig. 6: Non-Motion Compensation frame difference

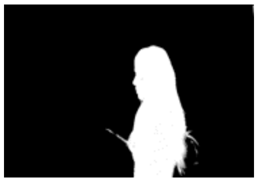


Fig. 7: Manually Matte

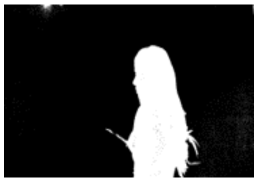


Fig. 8: Bayesian approach

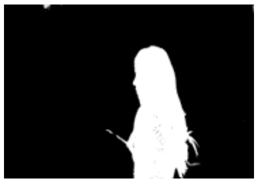


Fig. 9: 2D MRF

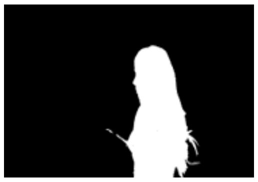


Fig. 10: 3D MRF

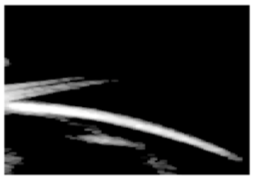


Fig. 11: Details of Fig(7)

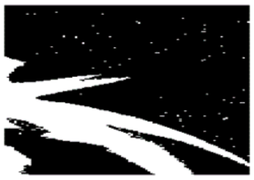


Fig. 12: Details of Fig(8)

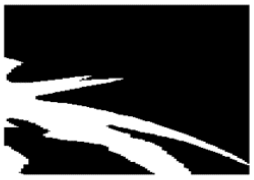


Fig. 13: Details of Fig(9)

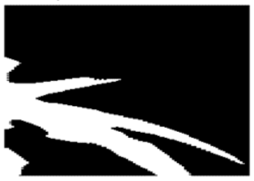


Fig. 14: Details of Fig(10)

- Figure(3) to Figure(6) illustrate the difference between motion compensation and Non-motion compensation.
- Figure(7) is the Manually Matte template. Figure(8) to Figure(10) illustrate the result of different approaches.

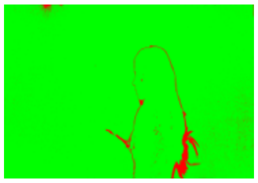


Fig. 15: Bayesian test

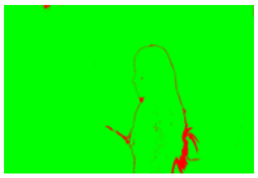


Fig. 16: 2DMRFtest

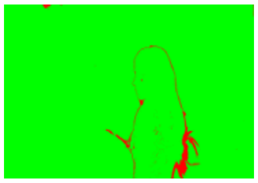


Fig. 17: 3DMRFtest

- Figure(15) to Figure(17) clearly illustrate the difference between the Manually matte and different approach

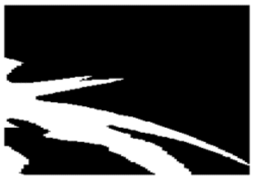


Fig. 18: 2D MRFICM

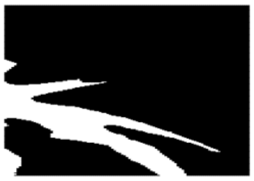


Fig. 19: 2D ICM details

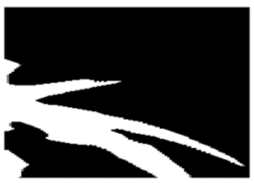


Fig. 20: 3D MRF details

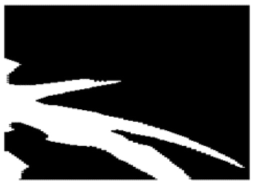


Fig. 21: 3D MRFICM details

- Figure(18) to Figure(21) shows the details after doing Iteration conditional modes and the original details before doing ICM.

The Markov random field is using the neighbourhood to get a better result in matting. So the key in this approach is the neighbourhood. If we using more neighbourhood number to help us to find out the α, we can have a better result. However, it also has it shortcomings. The more neighbourhood numbers mean larger calculation. We have to consider the balance between the cost of calculation and the quality of result. The 2D MRF approach is more fit for the imaging matting work and the 3D MRF is more fit for the video matting work. I compare just using one neighbourhood number in the 3D MRF approach for the previous frame and 8 neighbourhood numbers in the previous. The 8 neighbourhood result shows a better quality of the image matting than just one neighbourhood number.

VI. Improving the algorithm

There have some ways to improve this algorithm. First, we can consider the continuous three frame instead of just using the current frame and previous frame to do the 3D MRF. That is means we will consider the influence caused from the forward motions and also the influence from the backward motions. It must be more accurate. The simplest way to improve the algorithm is using more neighbourhood number in the 3D MRF, but it is not fit for the real movie industry compositing work.

References

[1] D. H. S. Yung-Yu Chuang, B. Curless and R. Szeliski, "A bayesian approach to digital matting," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition,CVPR 2001,Kauai, HI, USA*, pp. pp. II–II, 2001.

[2] C. R. Andrew Blake, Pushmeet Kohli, *Markov Random Fields for Vision and Image Processing*, 1st ed. Massachusetts,USA: MIT Press, 2011.

[3] T. Porter and T. Duff, "Compositing digital images," *In SIGGRAPH 1984*, pp. 253–239, July 1984.

[4] G. Winker, "Image analysis,random fields and markov chain monte carlo methods," *Springer Science Business Media, 2012*.

[5] J. W.Woods, "Markov random modeling," *IEEE Transactions on Automatic Control*, vol. 23, pp. 846–850, Oct 1978.

[6] S.Geman and D.Geman, "Stochastic relaxation,gibbs distribution,and the bayesian restoration of image," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, June 2007.

[7] E. M.F.Tappen, Ce Liu and W.T.Freeman, "Learning gaussian conditional random fields for low-level vision," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, June 2007.

[8] J. BESAG, "On the statistical analysis of dirty pictures," Ph.D. dissertation, May 7th,1986.

[9] A. Kokaram, *Motion Picture Restoration: Digital Algorithms for Artefact Suppression in Degraded Motion Picture Film and Video*, 1st ed. London,UK: Springer-Verlag, 1998.

[10] A. C. Kokaram, "Markov random fields (a rough guide)," *Electrical and Electronic Engineering Dept, University of Dublin, Trinity College*.

Appendix

Shida Sheng, Trinity College 17/03/2020

2D MRF Code

This is the code in nuke to implement the 2D Markov random field.

Listing 1: Nuke code

```
kernel mrf : ImageComputationKernel<eComponentWise>{
nuke.startPerformanceTimers
Image<eRead, eAccessRandom, eEdgeConstant > src; // the input image
Image<eRead, eAccessRandom, eEdgeConstant > l_energy; // the input image
Image<eWrite> dst; // the output
param:
float Lambda;
local:
float alpha;
void define() {
defineParam(Lambda, "Smothness Lambda",5.0f);
    }
void init(){
alpha = 25.0;
            }
void process(int2 pos) {
float left_pel = src(pos.x - 1, pos.y);
float left_pel_up = src(pos.x - 1, pos.y - 1);

float right_pel = src(pos.x + 1, pos.y);
float right_pel_up = src(pos.x + 1, pos.y - 1);

float top_pel = src(pos.x, pos.y - 1);

float bottom_pel = src(pos.x, pos.y + 1);
float bottom_pel_left = src(pos.x - 1, pos.y + 1);
float bottom_pel_right = src(pos.x + 1, pos.y + 1);

float e_sp_0_now = left_pel + right_pel + top_pel +bottom_pel + ↩
    left_pel_up + right_pel_up + bottom_pel_left + bottom_pel_right;

float e_sp_0 =  e_sp_0_now;

float e_sp_1_now = (1-left_pel) + (1-right_pel) + (1-top_pel) + (1-↩
    bottom_pel) + (1 - bottom_pel_left) +(1 - bottom_pel_right) + (1 - ↩
    left_pel_up) + (1 - right_pel_up);
float e_sp_1 = e_sp_1_now  ;

float e_0 = l_energy(pos.x,pos.y) + e_sp_0 * Lambda;
float e_1 = alpha + e_sp_1 * Lambda;
if(e_0 < e_1)
    dst() = 0;
if(e_0 > e_1)
    dst() = 1;
        }
        };
```

3D MRF Code

Listing 2: C Code.

```
kernel MRFKernel : ImageComputationKernel<eComponentWise>
{
Image<eRead, eAccessRandom, eEdgeClamped> src; // the input image
Image<eRead, eAccessRandom, eEdgeClamped> pre; // the input image
Image<eRead, eAccessRandom, eEdgeClamped> l_energy; // the likelihood ↩
    energy
Image<eRead, eAccessRandom, eEdgeClamped> l_energypre; // the likelihood ↩
    energy
//4 connected MRF with input energy

Image<eWrite> dst; //the output image

param:
float Lamda; // This parameter is made available to the user.

local:
float alpha;  // penalty for setting segmentation=1

void define() {
    defineParam(Lamda, "Smoothness Lamda", 20.0f);
  }
void init() {
    // 95% confidence interval
    alpha= 25.0f;
  }
//pos gives the positions of the kernel in the output images
```

```
  void process(int2 pos) {
  //current
float left_pel = src(pos.x - 1,pos.y);
float right_pel = src(pos.x + 1,pos.y);
float top_pel = src(pos.x,pos.y - 1);
float bottom_pel = src(pos.x,pos.y + 1);
float lefttop_pel = src(pos.x - 1,pos.y + 1);
float leftbottom_pel = src(pos.x - 1,pos.y - 1);
float righttop_pel = src(pos.x + 1,pos.y + 1);
float rightbottom_pel = src(pos.x + 1 ,pos.y - 1);
    //previous
float left_pel_pre =pre(pos.x - 1,pos.y);
float right_pel_pre=pre(pos.x + 1,pos.y);
float top_pel_pre=pre(pos.x,pos.y - 1);
float bottom_pel_pre=pre(pos.x,pos.y + 1);
float lefttop_pel_pre=pre(pos.x - 1,pos.y + 1);
float leftbottom_pel_pre=pre(pos.x - 1,pos.y - 1);
float righttop_pel_pre=pre(pos.x + 1,pos.y + 1);
float rightbottom_pel_pre=pre(pos.x + 1,pos.y - 1);

    // Calculate spatial energy for output=0(background),1(foreground)
float e_sp_0 = left_pel+right_pel+top_pel+bottom_pel+lefttop_pel+↩
   leftbottom_pel+righttop_pel+rightbottom_pel+  left_pel_pre+↩
   right_pel_pre+top_pel_pre+bottom_pel_pre+lefttop_pel_pre+↩
   leftbottom_pel_pre+righttop_pel_pre+rightbottom_pel_pre;
float e_sp_1 = (1-left_pel)+(1-right_pel)+(1-top_pel)+(1-bottom_pel)+(1-↩
   lefttop_pel)+(1-leftbottom_pel)+(1-righttop_pel)+(1-rightbottom_pel)↩
   +(1-left_pel_pre)+(1-right_pel_pre)+(1-top_pel_pre)+(1-bottom_pel_pre)↩
   +(1-lefttop_pel_pre)+(1-leftbottom_pel_pre)+(1-righttop_pel_pre)+(1-↩
   rightbottom_pel_pre);

float e_0 = l_energy(pos.x,pos.y) + l_energypre(pos.x,pos.y) + e_sp_0 * ↩
   Lamda;
float e_1 = alpha + e_sp_1 * Lamda;

if (e_0 < e_1 )
      dst() = 0;
if (e_0 > e_1 )
      dst() = 1;
  }
};
```