Automating Terrain Texturing in Real-Time Using a Rule-Based Approach

John Ferraris, Christos Gatzidis and Feng Tian

School of Design, Engineering and Computing, Bournemouth University, United Kingdom

***Abstract*— This publication proposes a novel approach to automatically colour and texture a given terrain mesh in real time. Through the use of weighting rules, a simple syntax allows for the generation of texture and colour values based on the elevation and angle of a given vertex. It is through this combination of elevation and angle that complex features such as ridges, hills and mountains can be described, with the mesh coloured and textured accordingly. The implementation of the approach is done entirely on the GPU using 2D lookup textures, delivering a great performance increase over typical approaches that pass colour and weighting information in the fragment shader. In fact, the rule set is abstracted enough to be used in conjunction with any colouring/texturing approach that uses weighting values to dictate which surfaces are depicted on the mesh.**

***Index Terms*—Terrain, Splatting, Real-Time Graphics.**

I. INTRODUCTION

Vast, real-time 3D terrains are used more and more in computer games today. The need for large and increasingly more complex landscapes requires an accompanying set of game content, such as terrain textures, needed to display the landscapes in a convincing manner. As a result, various techniques for automating and assisting the generation of such content have been explored over the years. As with other aspects of computing, consumer GPU hardware is becoming ever more powerful. Current-real time graphics techniques tend to favour placing significantly more of the workload on the GPU hardware, freeing up the CPU for other intensive game tasks, such as AI.

A significant amount of research has been conducted into the generation of terrain meshes. Procedural approaches are typical, as they can cater for specialised features (such as mountains and rivers) and offer a degree of control over the final output [1]. Work has gone into reducing the artefacts common to many terrain texturing approaches [2] and real world physical phenomena have been modelled to produce terrains with a high degree of realism [3].

Texture synthesis is another area that has found much use in terrain generation. By exposing a set of parameters to the artist, a near-infinite amount of terrain materials can be generated, adding greater realism to the scene and easing the burden of the artist ([4], [5], [6]). Research into generating seamless textures for arbitrary meshes at varying scales has offered solutions to the issue of transition artefacts and level of detail that are common obstacles to terrain generation and rendering ([7], [8], [9], [10]).

This publication will explore and implement an approach for the automatic assignment of terrain textures over an arbitrary mesh, according to a set of rules dictating how the basic textures should be blended together inside the GPU hardware. To demonstrate the result of this research and subsequent design, a fully interactive application for the generation/deformation of terrain meshes is produced.

Manuscript Received on 10 July 2010
E-mail: ftian@bournemouth.ac.uk

II. PREVIOUS RESEARCH

Smelik et al. [11] propose a declarative approach for terrain generation. As much as the algorithm succeeds in meeting the criteria of the case study, the abstraction process reduces the level of control over fine details which may make the technique ill-suited to applications outside this realm.

Roden and Parberry [12] also address the need for reducing content generation timeframes through automation. As the detailed geometry is computed on the GPU, the technique eases the burden of host memory and CPU cycles compared to [11], making it more suitable for general computer game applications which require processing power for other tasks.

De Carpentier and Bidarra [13] marry together the techniques of procedural terrain synthesis and interactive brush-based terrain sculpting. Unlike [11] and [12], the technique is perhaps not best suited to the generation of numerous, large scale terrains, but in contrast it does offer a higher degree of control over the finer details of the terrain geometry.

Roupé and Johansson [14] propose a technique for blending aerial photo textures with high resolution detail textures through the use of colour-coded image masks, allowing for the automation of content generation from a readily available source. The need for manual generation of the colour masks requires more artistic intervention than [12] and [11], although the extraction of masks from CAD drawings presents a solution to this problem.

Lai and Tai [15] present an offline approach to generating intermediately terrain textures to bridge the transitions between

different terrain types. Whilst the technique produces aesthetically-pleasing combinations from minimal user input, the amount of memory needed increases with texture resolution and the number of terrain types (each texture-to-texture transition requires 16 transition tiles).

Hardy and McRoberts [16] provide an approach similar to the rule set presented in this paper to assign splat texture weightings, which they call blending coefficients. They assign the weightings in the vertex shader where they were interpolated across the fragments in a manner typical of contemporary shader-based texture splatting. Our approach of using lookup tables offers a significant performance in frames per seconds benefit to this approach.

In this paper, based on some earlier work-in-progress development results [17], we will discuss a real time approach to automate the colouring and texturing an arbitrary terrain mesh. We also present a means of breaking up the uniformity of terrain colour using pseudo-random colour modulation. Our approach delivers frame-rates that are significantly greater than typical weight-interpolated texture splatting approaches.

III. WEIGHTING RULE SET

We propose a set of rules for defining the type of terrain that lies at a given vertex of the mesh according to the height and angle of that vertex. Through this simple rule set, complex and detailed terrains can be constructed automatically with smooth texture transitions without the need to manually paint the mesh. This combined rule set (elevation and angle) will be referred to throughout this document as the weighting rules. For the rest of this paper, we will discuss this rule set within the context of texture splatting [18], although the rules can be adapted for any terrain texturing technique.

3.1 Elevation Rules

The first part of the weighting rule set a vertex will go through is a fairly common technique for dictating the surface of the terrain according to the height of each vertex. We will refer to this technique as the elevation rules. In this occasion, the vertex's y component is checked against a set of predefined colour bands to determine what colour and texture weightings the vertex will be set to. The bands themselves lie within the 0 to 255 range as the mesh's height map will be sourced from an 8 bit greyscale texture. However, any desired range can be used according to the requirements of the application. Figure 1 illustrates an example set of five elevation bands: water (blue), sand (yellow), grass (green), rock (grey) and snow (white).

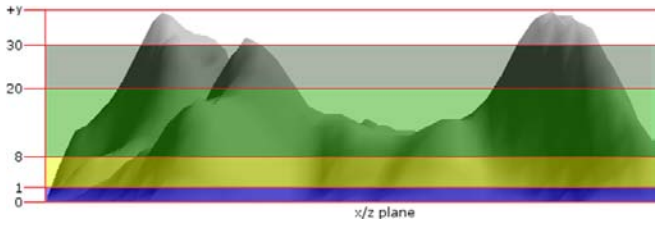


Fig. 1. Basic elevation bands (from lowest to highest): water, sand, grass, rock and snow.

In the figure above, each elevation band lies within a set height range. For example, the water band occupies a small band between 0 and 1 units in height, whereas the grass band occupies a larger range between 9 and 20. The full list of bands and their ranges is illustrated in Table 1 below. By assigning a vertex colour and splat weighting to each elevation band, simple terrains may be automatically generated without the need for artistic intervention.

TABLE 1. HEIGHT RANGES FOR EACH ELEVATION BAND.

Height	Colour/Splat Weighting
<= 1	Water
<= 8	Sand
<= 20	Grass
<= 30	Rock
31+	Snow

The results of the basic elevation rules prototype (Fig. 2) show significant signs of banding. This occurs because terrain features that span multiple elevation bands (such as hills and mountains) have their vertices coloured multiple times, resulting in an obvious striping effect, due to the colour difference where two bands meet. This can be reduced through the use of angle rules (to accurately colour and texture hills and mountains) and transition bands (to ease the abrupt transition between two elevation bands). Both of these techniques will be discussed in following sections.

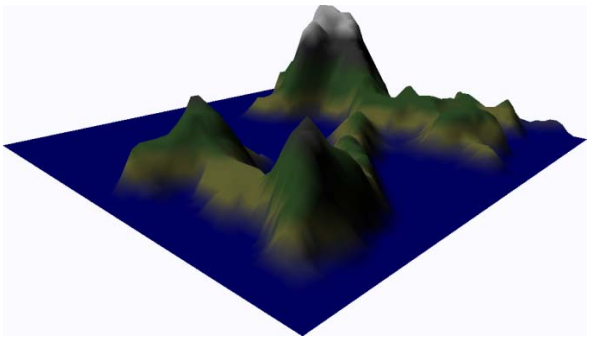


Fig. 2. Terrain colours according to basic elevation rules.

3.2 Transition Bands

Transition bands are intermediate elevation bands that colour and weigh a vertex as a mix of the two bands that the vertex lies between. The mix amount would be determined by the vertex's position within that band, e.g. closer to band 1 = heavy mix bias towards band 1, middle of band = 50/50 mix between band 1 and 2, closer to band 2 = heavy mix bias towards band 2. The transition bands can be implemented as separate, discrete bands (to allow for greater flexibility) or as a simple distance factor for each terrain band which determines the falloff of that particular terrain type from a predefined centre point within the band. For example, the grass band could have an elevation midpoint of 30, spanning +/- 10 units above and below this

point, falling off to a zero blend factor as the elevation reaches the upper and lower bounds of that band. Likewise, the sand and rock bands would have different midpoints but upper and lower bounds that overlap with the grass band, thus producing a linear gradient between the terrain types.

3.3 Colour Modulation

Another problem with the basic elevation rules (Fig. 2) is that there is no colour variation in the elevation bands due to the way that all of the vertices in the band are assigned the same stock colour. In order to break up the colour and add some variance, two or more colour variants could be specified for each band instead of one. Then, a linear interpolation of a random amount could be performed between the colour variants. The results of the colour modulation prototype show a marked improvement with regards to breaking up the colours in the elevation bands (Fig. 3). The effect is more apparent on the water and grass, where subtle variances give the illusion of a more complex terrain.

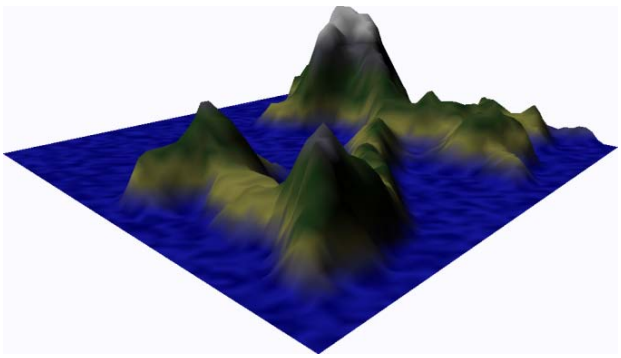
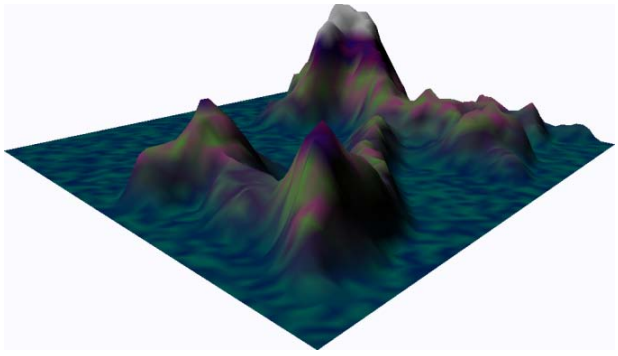


Fig. 3. Elevation rules with colour modulation break up bands of uniform colour.

Extreme variation between the two colours in an elevation band can yield exotic results (Fig. 4). Although not particularly convincing in terms of producing a photorealistic terrain, inventive use of this technique would allow for some flexibility with regards to developing "other-worldly" terrains. This, combined with an appropriate set of textures, could be used to create some truly exotic landscapes.



.

Fig. 4. Extreme colour modulation creates more exotic landscapes.

3.4 Angle Rules

The second part of the weighting rule set is the angle rules. These are additional rules that an elevation band can have that override the default colours/texture weightings of that band to colour/weight the vertex according to the angle of the vertex. For example, the grass elevation band could apply a mix of grass and rock if the angle of the vertex is greater than 30 degrees, instead of applying just the stock grass colour/texture. This allows for features such as hills and mountains in the grass elevation band to have their sides as a mix of rock and grass, with a bias towards rock the greater the angle of the vertex. The y component of a given vertex's normal is the angle of elevation in relation to the x/z plane. As the normal will be normalised, the y component will be in the 0 to 1 range, which, if inversed, equates to an angle range of 0 to 90 degrees. Table 2 below illustrates an example weighting rule set for three elevation bands, one of which has an additional angle rule subset:

TABLE 2. ANGLE RULES FOR TERRAIN TYPES.

Height	Angle	Colour/Splat Weighting
<= 20	All	Sand
<= 40	< 0.75	Rock
	>= 0.75	Grass
<= 60	All	Rock

The combination of these simple rules results in a terrain that is coloured in a more convincing manner (Fig. 5) than simply assigning a colour according to the vertex height. However, although the results could be greatly improved through further tweaking of the bands, simple colouring of the mesh does not demonstrate the full potential of the technique. Further results utilising texture splatting [18] will be displayed in the following sections.

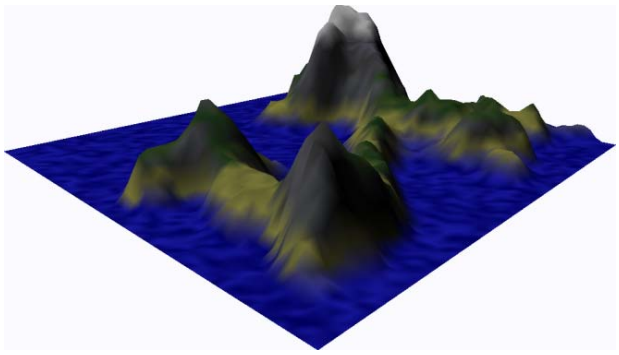


Fig. 5. Elevation and angle rules working together to break up the banding artifacts that occur when using the elevation rules on their own.（Color Plate 11）

IV. IMPLEMENTATION

For comparative purposes, two approaches are implemented: a typical interpolated approach (where the colours and splat weights are interpolated from the vertex shader across the fragments [16]) and a lookup approach (where the colours and splat weights are obtained in the fragment shader through use of

lookup textures). For each of these two techniques, the splatting section of the shader is identical, with the only difference being the input source of the splat weights and vertex colours.

The rule set for the interpolated approach is illustrated in Table 3 (below), where each terrain type (base, grass rock, sand and water) is represented by a pair of colours (initial and modulation) and a corresponding splat texture. The elevation and angle of each vertex are used as an interpolation factor for the terrain types, for example the sand/grass band used the elevation within that band to dictate the amount of sand and grass for a given vertex. The colour modulation is implemented through a random linear interpolation between the initial and modulation colour for a given terrain type. The rule set is executed on the CPU and the resulting colours and splat weights are included as attributes for each vertex in the terrain mesh. The frame to frame coherence is maintained as the modulated colour of each vertex is stored directly as part of the mesh vertex data.

TABLE 3. WEIGHTING RULES USED FOR THE INTERPOLATED APPROACH.

Height	Angle	Colour/Splat Weighting
<= 1	All	Water
<= 8	>= 0.5	Sand/Water
	< 0.5	Sand/Rock
<= 20	>= 0.5	Sand
	< 0.5	Sand/Rock
<= 30	>= 0.75	Sand/Grass
	< 0.75	Rock/Grass
<= 255	>= 0.75	Grass
	< 0.75	Grass/Base
	< 0.7	Rock/Grass

4.1 Texture Splatting

To simplify the generation of the elevation and angle lookup texture, a combination of 4 low resolution splat textures (and 1 base texture) along with accompanying high resolution detail textures are used to texture the mesh. The detail textures are faded out over a user-definable cut-off distance to enable smooth blending between upfront mesh faces (with high-detailing) and further back faces (with no detailing).

The textures are mapped onto the terrain mesh using texture splatting [18] by using a set of low resolution splat textures and an associated set of high resolution detail textures. Texture splatting is chosen as it is simple to implement and demonstrates the weighting rule set well. The combination of using low resolution and high resolution textures allows for a level of detail algorithm to switch between high and low resolution textures and solitary low resolution textures, depending on the distance of the mesh primitives from the viewer position.

In addition to the splat/detail textures, a series of blending weights in the 0 to 1 range are used for each vertex in order to determine how much of each splat/detail combination will be blended onto the mesh.

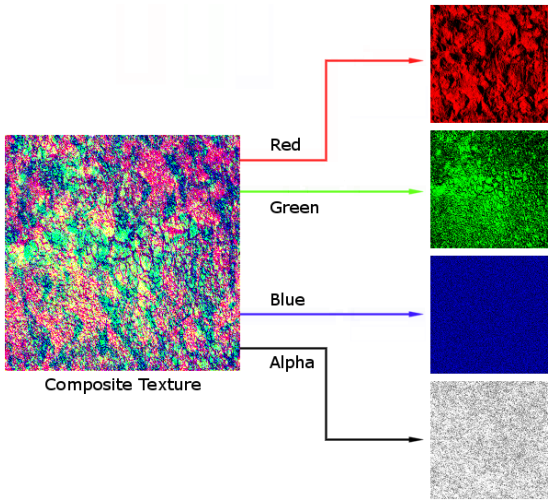


Fig. 6. Multiple detail textures are compounded into a single texture, where each detail texture occupies a single colour channel.

A set of suitably low resolution textures are used as the basic splat textures. Any resolution would be permitted, however, a correct balance between the resolution of the splat and detail textures is needed to maintain optimum performance. Therefore, for this application, the splat textures measure 32x32 pixels. This allows for enough texel data to create varied textures as the splats are linearly interpolated across the mesh without being a high enough resolution to negate the effect of the detail textures.

For each splat texture, an associated high resolution texture is blended on top of the splat texture to provide high level of detail when viewed up close. Any resolution will be permitted, although the chosen resolution should provide a good balance between not being too large but providing enough resolution to give the illusion of detail on the terrain mesh. The detail textures are greyscale, thus allowing for up to four detail textures being stored in a given composite RGBA texture (shown in Fig. 6).

Each of the five splat textures are combined in the fragment shader according to a set of four splat weights for each vertex, as calculated by the weighting rules. The reason for four weightings, as opposed to five, is due to the order in which the splat textures are combined.

The rest of this section will be concerned with the implementation of the lookup approach.

4.2 Texture Splatting

The elevation and angle rules are implemented as a 2D lookup texture with the x axis representing the vertex angle and the y axis representing the vertex elevation (Fig. 7). Note that this figure does not represent the actual lookup texture used to generate the images in this publication; rather the values used to populate the texture have been embellished for illustrative purposes.

Any size and format can be used depending on the required accuracy and VRAM footprint. For lower bit depths, noise would be used to break up banding artifacts due to the limited value range of each colour channel. For our implementation we choose a 256x256 texture in 32-bit floating point RGBA format for simplicity and accuracy. This however limited the number

of splats used to 5 (one for each colour channel of the lookup texture with the base splat weight implied by the splat blending equation in Fig. 8), although adapting the approach to accommodate the 8 splats as originally intended could be achieved by compressing 2 splat weights into a single colour channel using a suitable data format.

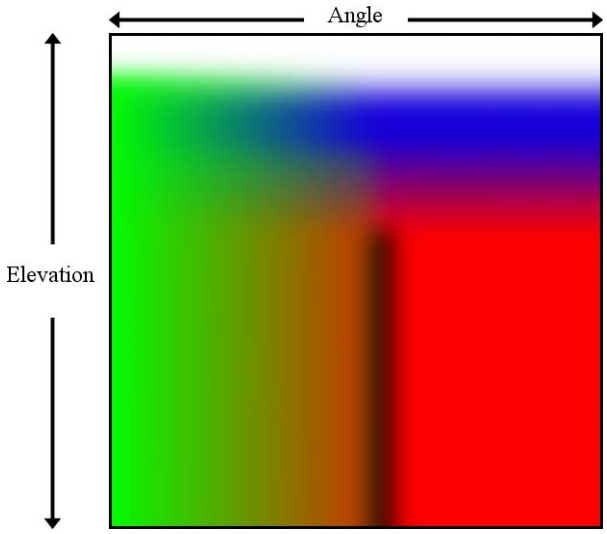


Fig. 7. The elevation and angle lookup texture for the 5 terrain textures. The vertical black bar represents a brief transition of dirt (the base texture) between rock and grass on features such as mountain plateaus and cliffs.（Color Plate 12）

The UV coordinates used to perform texture lookup within the fragment shader are obtained from the y component of the vertex normal and the scaled y component of the vertex position. Once the lookup has been established, the resulting 4 part colour component is used to perform the blending of the terrain splats, with the red component dictating the grass weighting, the green component the rock weighting, the blue component the sand weighting and the alpha component (shown as white in the figure above) the water weighting. These values are then plugged into the splat blending equation in Listing 1, with the base splat amount being dictated by a grass weighting of less than 1.

```
result = mix(base, grass, lookup.r);
result = mix(result, rock, lookup.g);
result = mix(result, sand, lookup.b);
result = mix(result, water, lookup.a);
```

Listing. 1. The splat blending equation. The base texture is implied by a lack of grass (e,g, a grass weighting of less than 1).

4.3 Colour Modulation

The subtle colour modulation used to break up the uniform colour of the terrain types is implemented through a 2D texture lookup table, with the terrain type proceeding down the y axis and the modulation colours interpolated along the x axis (with the initial and modulation colour at opposite ends of the axis, show in Fig. 8).

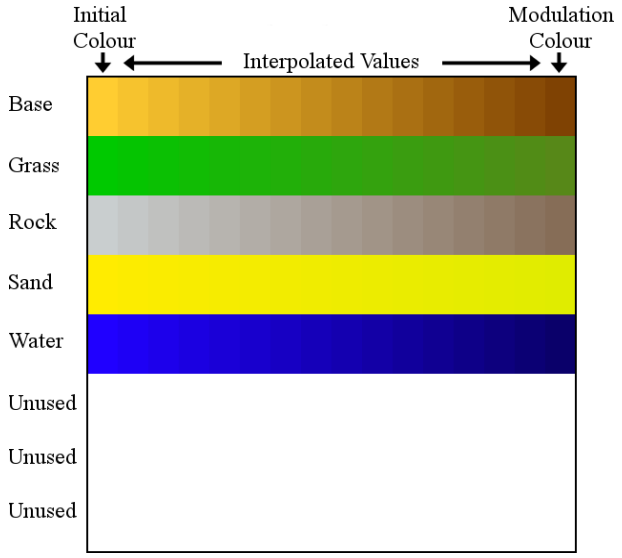


Fig. 8. The colour lookup texture for the full 8 splats, of which only 5 were used.

The colour lookup texture is created upon application start up and is updated on demand as and when the user changes one or more of the terrain type colours. The image height needs to be a multiple of the number of splats (in this case 8, although only 5 are actually used) and the width can be any width that conforms to the GPU's allowed dimensions and the desired number of interpolated colour variations. In our case, we choose a 32-bit RGBA image with a width and height of 16 pixels, the minimum size required by our development GPU.

```
for(int y = 0; y < height; y++)
{
   step = width / numSplats;
   col  = splats[(int)(y / step)].initial;
   mod  = splats[(int)(y / step)].mod;

   for(int x = 0; x < width; x++)
   {
      int index = y * width + x;
      out = lerp(col, mod, x / width);
      img[index * 4 + 2] = out.r * 255;
      img[index * 4 + 1] = out.g * 255;
      img[index * 4] = out.b * 255;
      img[index * 4 + 3] = 255;
   }
```

Listing. 2. The pseudo-code for generating the colour lookup table texture.

No texture filtering is used as the 16 colour variations produce a wide enough variety to break up the uniformity of colour.

The code in Listing 2 (above) illustrates how this colour lookup texture is generated. The lookup table is then accessed in the fragment shader where upon the UV coordinates used to

perform the lookup are derived from mapping the splat type to the V component and the desired modulated colour by assigning a pseudo-random number to the U component.
This pseudo-random number is an attribute of each vertex in the mesh that is randomly generated for each vertex at the mesh construction stage. This allows for a random lookup value for each vertex whilst maintaining frame to frame coherence as, being part of the mesh vertex data, these values are persistent between each frame.

V. RESULTS

Fig. 10 shows the output when the weighting rules are combined with the texture splatting technique. Through just 4 simple terrain textures and a modest rule set, a rocky outcrop within a body of water is generated from a height map without any artist intervention.

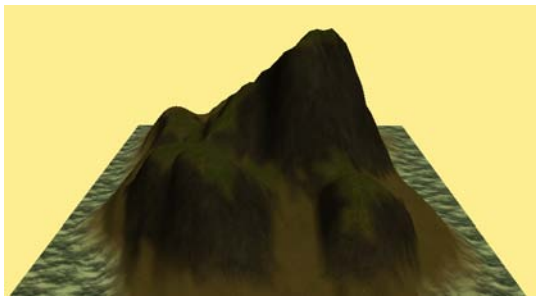


Fig. 10. Texture splatting combined with the weighting rules. This terrain was generated without any artistic intervention.

Fig. 11 illustrates the elevation rules for the final elevation band, grass. This is the most complicated of the elevation bands, as it transitions between grass, rock and the base texture. The use of three transitions, rather than the typical two used in the other elevation bands, causes hills and mountains to transition between the grass texture for moderate slops/peaks/plateaus and rock/grass for ridges/mountains in the usual manner, but the third transition “rounds off” hills by performing a small base splat transition between the rock and grass transition.

This produces the effect of the edges of slops/peaks/plateaus to have an earth-like colour, much as the edges of slops/peaks/plateaus would do in real life.

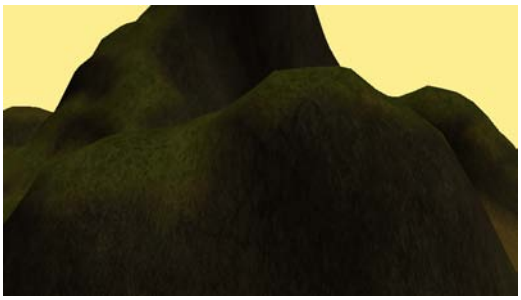


Fig. 11. Angle rules for the grass band. Notice the subtle brown round-off at the transition between rock and grass. This is due to a narrow band of dirt texture between the rock and grass angle rules. This is illustrated as the black vertical bar in Fig. 7.

The angle rules for this band dictate that moderate slopes/plateaus are to be coloured/weighted as grass. A small transition band blends between grass and the base texture, with the blend amount being the angle of the vertex (the black area in Fig. 7). This causes a bias towards grass with an approximate 3 to 1 ratio in the favour of the grass. The final transition band is between rock and grass for steep surfaces, with the weighting between rock and grass being the angle fractionalised heavily towards grass. This causes the more dramatic angles such as steep cliff faces to be heavily biased in favour of rock, whereas more moderate angles are in favour of grass. A slight, muddy transition between the grass and the rock occurs due to the narrow angle band that the grass/base transition operates at, causing the edges of rocks and hills to be coloured with a dirt hue before dropping off into the rock.

Fig. 12. Elevation and angle rules for the sand/grass transition band. In this instance, the steepness of the rock feature has overridden the elevation rules for this particular band. The rock feature in the background illustrates how as the feature plateaus, the usual elevation band (in this case grass) regains dominance.

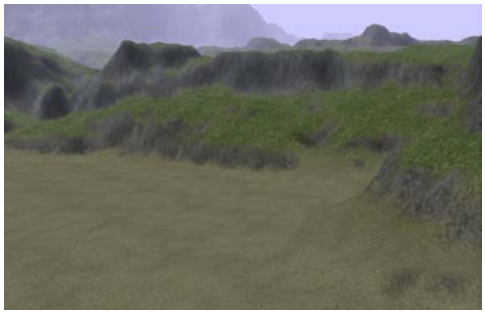


Fig. 13. An example of a final terrain model produced by the implementation. This terrain is larger in scale than the ones in previous figures and features mostly grass and sand.

Fig. 12 illustrates the sand/grass transition band. As the terrain erupts abruptly from the relatively level sand band, rocky features appeared to protrude from the sand.

This overrides the typical sand/grass transition, although in the background one can see this transition as one of the rocky features trails down in gradient, creating the illusion of a grassy trail leading down from the rock into the sand.

VI. PERFORMANCE ANALYSIS

Table 4 shows the performance results for the interpolated and lookup approach when brute-force-rendering a 513x513 terrain mesh with 5 splats.

The reason for 513x513 vertices is due to our implementation taking a texture of such dimensions to produce a mesh of 51x512 quads, where each quad is made up of four vertices.

This particular nuance has no bearing on how the rule set is implemented or operates; rather it was a decision independent from the techniques described in this paper.

The varying parameters in the third column represent values that are passed and interpolated from the vertex shader to the fragment shader.

The lookup approach renders with approximately a 41% increase in frames per second. Although the approach requires more texture lookups in the fragment shader, the performance increase can be attributed to the fewer varying parameters calculated in the vertex shader and interpolated across the fragments.

This reduces the amount of per-fragment interpolation and fetches in the vertex stream as well as fetches and assignments in the fragment shader.

Contemporary GPUs can perform fragment shader texture lookups very efficiently so offloading operations to texture lookups can give significant performance increases.

TABLE 4. PERFORMANCE RESULTS FOR A 513X513 TERRAIN MESH (ATI RADEON HD 4600). THE 41% INCREASE IN FRAME-RATE IS DUE TO THE CHEAPNESS OF TEXTURE LOOKUPS COMPARED TO THE RELATIVE EXPENSE OF VARYING PARAMETER INTERPOLATION.

Approach	FPS	No. of varying parameters (in floats)	No. of texture lookups
Interpolated	135	13	7
Lookup	190	7	13

VII. CONCLUSION AND FUTURE WORKS

Although the implementation presented here uses a modest texture and rule set combined with basic blending techniques, the weighting rules could be used in conjunction with more contemporary texturing techniques, such as normal mapping and blend masking [16].

Even just expanding the texture and rule set along with using higher quality, more detailed textures would allow for far more convincing and detailed terrains.

Furthermore, the technique is not restricted to splat textures. For example, the rule set could be used to dictate how a set of procedural textures are applied to a terrain mesh.

In fact, any approach based on the angle and elevation of a surface can benefit from this technique.

The full implementation of 8 splats is reduced to 5 to simplify artistic generation of elevation and angle lookup texture so the algorithm is not tested and measured under full load.

It may transpire that the performance increase from fast fragment-based lookup tables drops enough as the number of splat increases to a level that offers no significant advantage over the typical approach where the blend weights and colours are stored in the vertex data and interpolated across the fragments.

If this is the case, an alternative might be to bake the blend weights and vertex colours into separate textures of the same dimensions as the terrain's height map.

These textures could be updated on-demand as and when the user changes the elevation of the terrain using a modification of the rule set shaders with these textures as the render targets.

As such targets have yet to reach maturity in terms of speed, such operations may need to be batched in order to maintain acceptable frame rates, especially for operations that affect a broad section of large terrain meshes

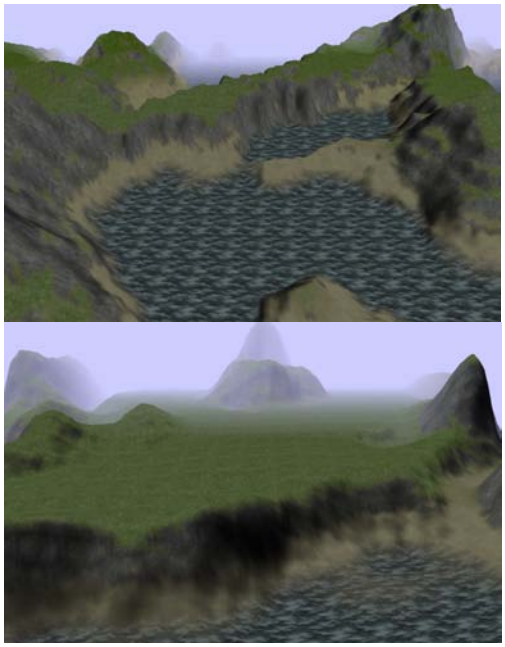


Fig. 14. Examples of arbitrary terrain texturing using worldly texture sets.

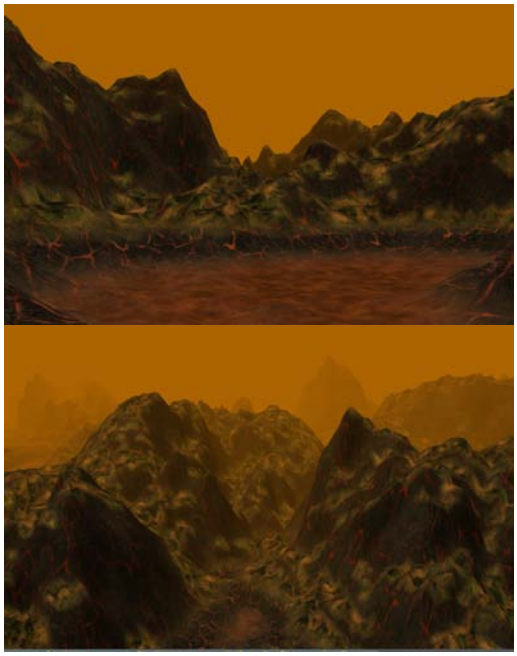


Fig. 15. Examples of arbitrary terrain texturing using worldly texture sets.

REFERENCES

[1] K. Raiyan Y. Kamal, U. Sarwar,, "Parametrically Controlled Terrain Generation", *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, ACM, Perth, Australia, 2007, pp. 17-23

[2] Mc Roberts, A. Hardy, "Level of Detail for Terrain Geometry Images", *Proceedings of the 5th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, ACM, Grahamstown, South Africa, 2007, pp. 25-30

[3] O. Stava, B. Benes, M. Brisbin, O. Krivanek, "Interactive Terrain Modeling Using Hydraulic Erosion", *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association , Dublin, Ireland, 2008, pp. 201-210

[4] J. Rhoades, G. Turk, A. Bell, A. State, U. Neumann, A. Varshney, "Real-Time Procedural Textures", *Proceedings of the 1992 symposium on Interactive 3D graphics*, ACM, Cambridge, Massachusetts, United States, 1992, pp. 95-100

[5] S. Lefebvre, N. Neyret, "Pattern Based Procedural Textures", *Proceedings of the 2003 symposium on Interactive 3D graphics*, ACM, Monterey, California, 2003, pp. 203-212

[6] S. Lefebvre, H. Hoppe, "Parallel Controllable Texture Synthesis", *ACM Transactions on Graphics*, 24(I3), ACM, Los Angeles, California, July 2005, pp. 777-786

[7] F. Neyret, M-P. Cani, "Pattern-Based Texturing Revisited", *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co, 1999, pp. 235-242

[8] J. Zhang, K. Zhou, L. Velh, G. Baining, H-Y. Shum, "Synthesis of progressively-variant textures on arbitrary surfaces", *ACM Transactions on Graphics*, 22(I3), ACM, San Diego, California, July 2003, pp. 295-302

[9] C. Guo, S-C. Zhu, Y. Wu, "Visual Learning by Integrating Descriptive and Generative Methods", *Department of Statistics Papers*, Los Angeles, 2000

[10] O. Wu, Y. Yu, "Feature Matching and Deformation for Texture Synthesis", *ACM Transactions on Graphics*, 23(I3), ACM, Los Angeles, California, August 2004, pp. 364-367

[11] R. M. Smelik, T. Tutenel, K. J. de Kraker, R. Bidarra, "Declarative terrain modeling for military training games", To be published in: *International Journal of Computer Games Technology*, 2010

[12] T. Roden, I. Parberry, "From Artistry to Automation: A Structured Methodology for Procedural Content Creation", *Department of Computer Science & Engineering*, University of North Texas, 2004.

[13] G. De Carpentier, R. Bidarra, "Interactive GPU-Based Procedural heightfield Brushes", *Proceedings of the 4th International Conference on Foundations of Digital Games*, ACM, Orlando, Florida, 2009, pp. 55-62

[14] M. Roupé, M. Johansson, "Visual quality of the ground in 3D models: Using color-coded images to blend aerial photos with tiled detail textures", *Proceedings of the 6th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, ACM, Pretoria, South Africa, 2009, pp. 73-79

[15] Y-Y. Lai, W-K. Tai, "Synthesizing Transition Textures on Succession Patterns", *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, ACM, Dunedin, New Zealand, 2005, pp. 273-276

[16] A. Hardy, D. McRoberts, "Blend Maps: Enhanced Terrain Texturing", *Proceedings of the 2006 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries*, V24, South African Institute for Computer Scientists and Information Technologists , Somerset West, South Africa, 2006, pp. 61-70

[17] J. Ferraris, C. Gatzidis, "A Rule-based Approach to 3D Terrain Generation via Texture Splatting", *5th International Conference For Advances In Computer Entertainment*, ACM Press, Athens, Greece, 2009, pp. 407-408

[18] C. Bloom. *Terrain Texture Compositing by Blending in the Frame-Buffer*. 2000 http://www.cbloom.com/3d/techdocs/splatting.txt

John Ferraris is a PhD researcher at the School Of Design, Engineering & Computing in Bournemouth University (BU), UK. He received his BSc (Hons) in Computing at Bournemouth University in 2009. His research interests include real-time 3D graphics, terrains, lighting and texturing.

Christos Gatzidis is a lecturer at the School Of Design, Engineering And Computing at Bournemouth University. He has received his PhD from City University London and has published research work in various international conferences. He has also been a member of several international program committees. His research interests include, amongst others, games technology, non-photorealistic rendering, mobile navigation and serious games.

Feng Tian is a Senior Lecturer at the School Of Design, Engineering & Computing in Bournemouth University (BU), UK. He has received his PhD from Xi'an Jiaotong University, China and worked as Assistant Professor in Nanyang Technological University, Singapore, prior to joining BU. He's been researching and developing novel ideas and technologies for more than 10 years in the field of computer animation, graphics, augmented reality, NPR, etc.