AMF Tutorial: Colors and Textures (Part 2)

Hod Lipson[1]

Abstract

This tutorial provides an introduction to the specification of colors and color textures in the AMF language. This is the second of a series of tutorials that cover geometry, graded materials, colors, textures, lattices, curved triangles, constellations, and other AMF features.

When 3D printing was invented back in the 1980s, you could print in any color you wanted, as long as it was a pale yellow tint. Stereolithography was the dominant technology at the time, and the only material it could print with was a photo-curable polymer that solidified into a yellowish resin. Other technologies that followed, such as fused deposition modeling (FDM) and selective laser sintering (SLS), also used polymers with only a single flat color option. A few years later, alternative color options became available, but one could still print in only a single color at a time. The burgeoning industry focused on improving resolution, strength, and functionality of printed parts, and their color was a distant secondary afterthought for a long time. When Z-Corp introduced color 3D printing in the early 2000s for the first time, it came at the expense of functionality, leading to 3D-printed models that were beautifully colored, but useful for visualization only.

With the advent of multimaterial 3D printing today, however, the importance of color printing has come to the forefront. Even some of the relatively low-cost, consumer-scale 3D printers can print in multiple materials simultaneously. One can 3D print an object comprising multiple assembled subcomponents in a single print job, each subcomponent printed in a different color. More importantly, one can vary the color of the printed part gradually, creating shades of colors using just a few primary colors. It is even possible to 3D print complete images, in full 24-bit color.

Those used to working with STL files cringe at the thought of printing in color, since STL files do not have any accommodation for color information. Over the years, 3D printers that could handle color in some fashion accommodated that information using a hodgepodge of ad-hoc solutions. One solution involved the use of multiple STL files, one for each color. That approach does not work well for graded colors and images. Others adopted the use of VRML files, but that format was originally intended for graphic display only, and so it does not ensure that an object is printable.

The AMF file format[1] deliberately targets color 3D printing and allows the explicit specification of colors in a variety of ways. One can easily specify the color of the entire object, or specify individual colors for separate volumes. It is possible also to associate colors with distinct materials. It is also easily possible to specify gradual color transitions, and even complete texture maps (Fig. 1).

This tutorial will demonstrate the use of colors in AMF. We assume a basic knowledge of the AMF format, as covered in AMF tutorial 1.[2] We start with a simple tetrahedron defined using four vertices and four triangles, using a fairly self-explanatory AMF file as shown in Figure 2. All files used in this tutorial and the AMF viewer are available on the AMF wiki.[3]

Coloring an Object or Volume with a Solid Color

In AMF, colors are specified using either the `<color>` element or the `<texmap>` and `<texture>` elements. The `<color>`

[1]Cornell University, Ithaca, New York.

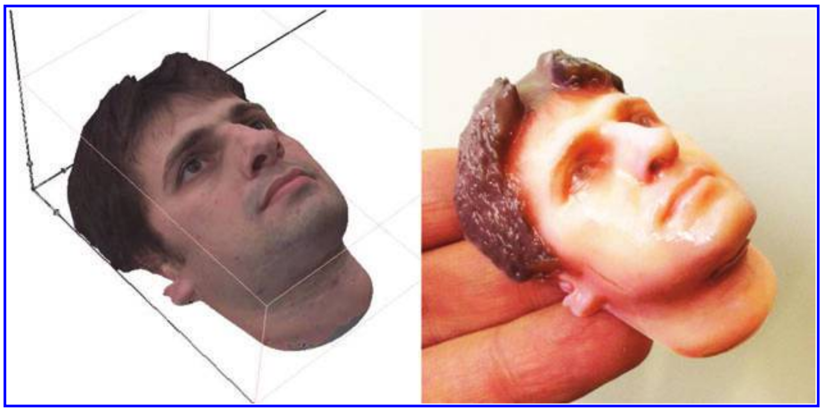
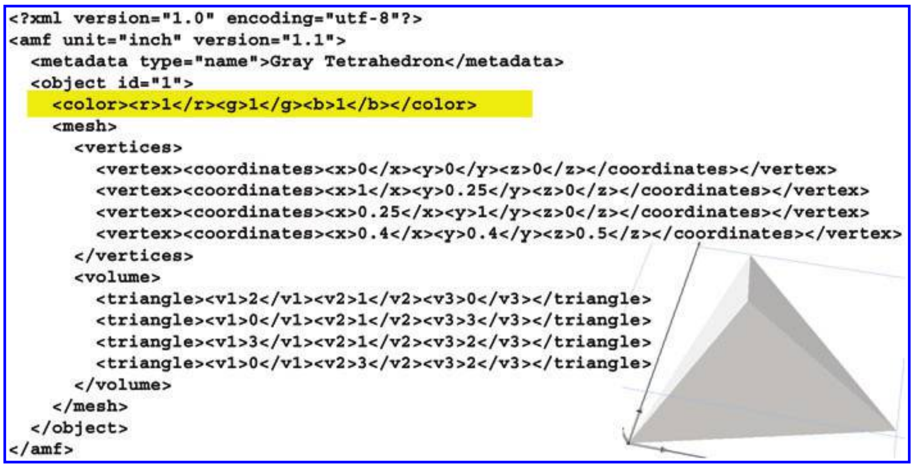
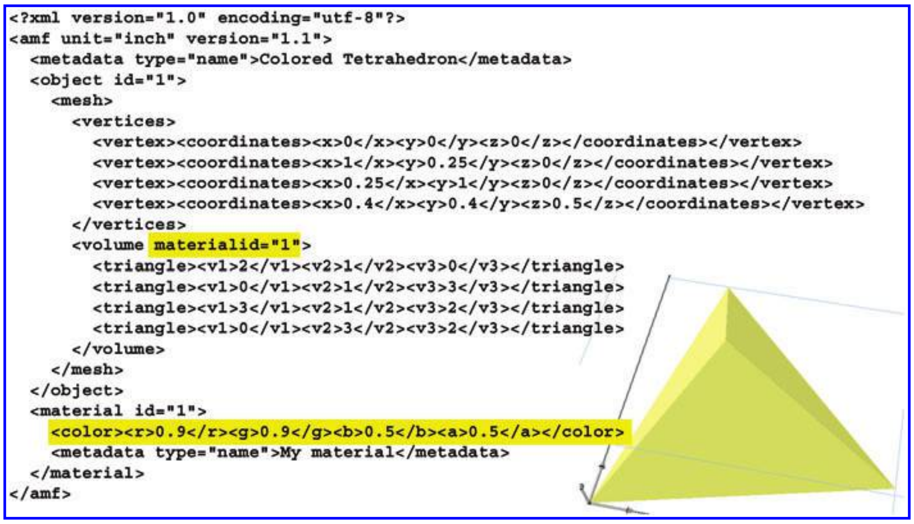


Figure 1. A full color texture AMF file in viewer (*left*) printed directly on a CONNEX3 (*right*). File available on AMF Wiki,[2] courtesy of Daniel Dikovsky, Stratasys. Color images available online at www.liebertpub.com/3dp



```
<?xml version="1.0" encoding="utf-8"?>
<amf unit="inch" version="1.1">
  <metadata type="name">Gray Tetrahedron</metadata>
  <object id="1">
    <color><r>1</r><g>1</g><b>1</b></color>
    <mesh>
      <vertices>
        <vertex><coordinates><x>0</x><y>0</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>1</x><y>0.25</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>0.25</x><y>1</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>0.4</x><y>0.4</y><z>0.5</z></coordinates></vertex>
      </vertices>
      <volume>
        <triangle><v1>2</v1><v2>1</v2><v3>0</v3></triangle>
        <triangle><v1>0</v1><v2>1</v2><v3>3</v3></triangle>
        <triangle><v1>3</v1><v2>1</v2><v3>2</v3></triangle>
        <triangle><v1>0</v1><v2>3</v2><v3>2</v3></triangle>
      </volume>
    </mesh>
  </object>
</amf>
```

Figure 2. Default color specification in AMF. Here the default color for object "1" is set to gray. File available on AMF Wiki.[3] Color images available online at www.liebertpub.com/3dp



```
<?xml version="1.0" encoding="utf-8"?>
<amf unit="inch" version="1.1">
  <metadata type="name">Colored Tetrahedron</metadata>
  <object id="1">
    <mesh>
      <vertices>
        <vertex><coordinates><x>0</x><y>0</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>1</x><y>0.25</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>0.25</x><y>1</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>0.4</x><y>0.4</y><z>0.5</z></coordinates></vertex>
      </vertices>
      <volume materialid="1">
        <triangle><v1>2</v1><v2>1</v2><v3>0</v3></triangle>
        <triangle><v1>0</v1><v2>1</v2><v3>3</v3></triangle>
        <triangle><v1>3</v1><v2>1</v2><v3>2</v3></triangle>
        <triangle><v1>0</v1><v2>3</v2><v3>2</v3></triangle>
      </volume>
    </mesh>
  </object>
  <material id="1">
    <color><r>0.9</r><g>0.9</g><b>0.5</b><a>0.5</a></color>
    <metadata type="name">My material</metadata>
  </material>
</amf>
```

Figure 3. Color associated with a material. Here a translucent yellow-gray color is assigned to material "1." Any volume associated with that material will automatically have that color, unless overridden by vertex or triangle color specifications. File available on AMF Wiki.[2] Color images available online at www.liebertpub.com/3dp

element is typically used for solid colors or for graded colors, whereas the `<texmap>` element is used for full 2D or 3D texture mapping.

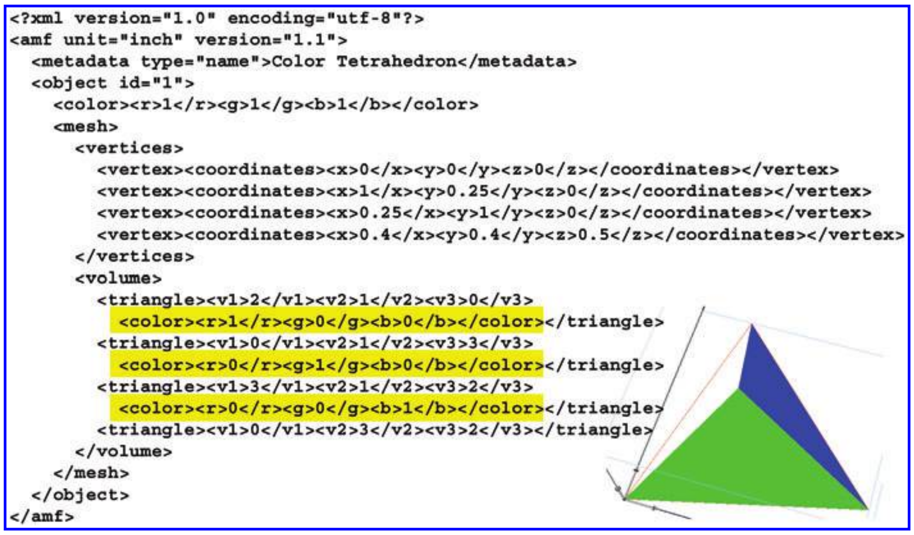
The color element takes the form `<color><r>R</r><g>G</g><b>B</b><a>A</a></color>`, where R, G, and B represent the level of red, green, and blue color components as numbers between zero and one. The alpha channel (transparency) can also be specified as the number A.

Let's begin with a simple example. If you don't specify any color information at all in your AMF file, the importing program or viewer will need to assume some default color, as they do today with STLs. Most software packages default to a grayish color, while others pick a random bright color for you. If you want to control that default color, insert a `<color>` element in the beginning of each `<object>` element. Figure 2 shows an example of inserting a color element that specifies that object "1" defaults to white. A similar color statement could be inserted into the beginning of each `<volume>` element. If an object comprises multiple volumes, each volume can be colored differently. This is particularly useful for highlighting one of the volumes or color coding subcomponents.

A more physically realistic way to specify the colors of an object or its components is to assign a material to each volume, and to assign a color to the material. That way the color of the volume is specified indirectly, because the color of the volume becomes the color of the material that is assigned to that volume. For example, Figure 3 assigns a translucent yellow color to material "1," and then assigns material "1" to the first object, thereby making it yellow.

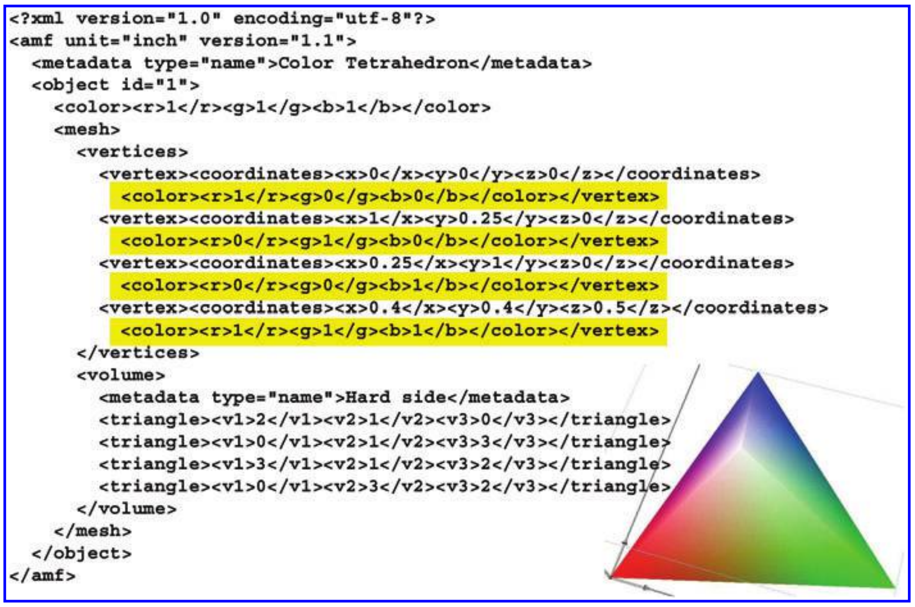
Coloring an Object or Volume with Multiple and Graded Colors

Sometimes objects need to be composed of multiple colors that vary smoothly or abruptly in some pattern. A simple way to do this is to assign colors to specific



```
<?xml version="1.0" encoding="utf-8"?>
<amf unit="inch" version="1.1">
  <metadata type="name">Color Tetrahedron</metadata>
  <object id="1">
    <color><r>1</r><g>1</g><b>1</b></color>
    <mesh>
      <vertices>
        <vertex><coordinates><x>0</x><y>0</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>1</x><y>0.25</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>0.25</x><y>1</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>0.4</x><y>0.4</y><z>0.5</z></coordinates></vertex>
      </vertices>
      <volume>
        <triangle><v1>2</v1><v2>1</v2><v3>0</v3>
          <color><r>1</r><g>0</g><b>0</b></color></triangle>
        <triangle><v1>0</v1><v2>1</v2><v3>3</v3>
          <color><r>0</r><g>1</g><b>0</b></color></triangle>
        <triangle><v1>3</v1><v2>1</v2><v3>2</v3>
          <color><r>0</r><g>0</g><b>1</b></color></triangle>
        <triangle><v1>0</v1><v2>3</v2><v3>2</v3></triangle>
      </volume>
    </mesh>
  </object>
</amf>
```

Figure 4. Colors associated with specific triangles override default object or material color. The color of one of the triangles here was left unspecified, so it defaulted to the object color (white). Color images available online at www.liebertpub.com/3dp



```
<?xml version="1.0" encoding="utf-8"?>
<amf unit="inch" version="1.1">
  <metadata type="name">Color Tetrahedron</metadata>
  <object id="1">
    <color><r>1</r><g>1</g><b>1</b></color>
    <mesh>
      <vertices>
        <vertex><coordinates><x>0</x><y>0</y><z>0</z></coordinates>
          <color><r>1</r><g>0</g><b>0</b></color></vertex>
        <vertex><coordinates><x>1</x><y>0.25</y><z>0</z></coordinates>
          <color><r>0</r><g>1</g><b>0</b></color></vertex>
        <vertex><coordinates><x>0.25</x><y>1</y><z>0</z></coordinates>
          <color><r>0</r><g>0</g><b>1</b></color></vertex>
        <vertex><coordinates><x>0.4</x><y>0.4</y><z>0.5</z></coordinates>
          <color><r>1</r><g>1</g><b>1</b></color></vertex>
      </vertices>
      <volume>
        <metadata type="name">Hard side</metadata>
        <triangle><v1>2</v1><v2>1</v2><v3>0</v3></triangle>
        <triangle><v1>0</v1><v2>1</v2><v3>3</v3></triangle>
        <triangle><v1>3</v1><v2>1</v2><v3>2</v3></triangle>
        <triangle><v1>0</v1><v2>3</v2><v3>2</v3></triangle>
      </volume>
    </mesh>
  </object>
</amf>
```

Figure 5. Colors associated with specific vertices override default object or material color. Color is interpolated between vertices, creating a graded color effect. Color images available online at www.liebertpub.com/3dp

triangles or vertices of the mesh. When colors are assigned to triangles, then the entire triangle assumes a solid color. Figure 4 shows an example of a tetrahedron with three of its four triangular faces colored in red, green, and blue. The fourth facet, shown on the left side of the inset picture in Figure 4, is shown in white. Since no color was specified for the fourth facet, the default white color specified at the beginning of the `<object>` element was assumed.

A color specification hierarchy determines how conflicting color specifications are handled. A local color specification always overrides a global color specification. At the lowest rung of the hierarchy is the file-wide, most global color setting specified at the beginning of the `<amf>` element. Next up are colors specified at the beginning of each object, as a direct color or by association with a colored material. Then come colors specified for individual volumes, triangles, and finally vertices.

Figure 5 shows coloring at the vertex level. A different color is specified for each vertex of the object. Areas in between vertices are interpolated using Gouraud shading.

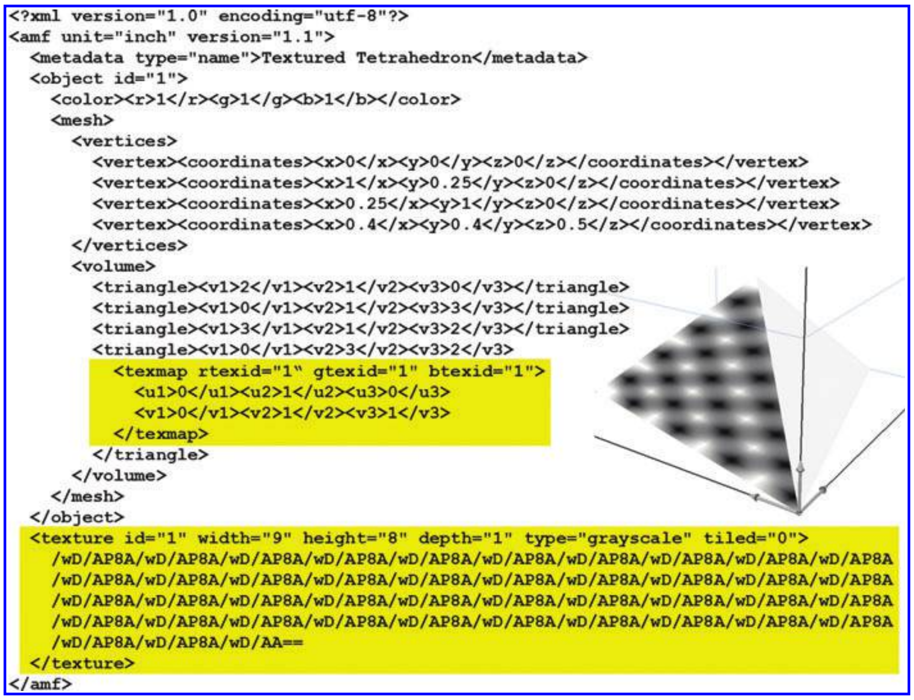
Color Texture Mapping

The most advanced form of color specification involves texture mapping. To accomplish texture mapping, we first include a new `<texture>` element that defines the image to be mapped. That image is essentially a bitmap image, defined as an array of grayscale levels between 0 and 255. The bitmap can be of any size, as specified by the `<texture>` attributes. For example, the texture command in Figure 6 specifies a grayscale bitmap of size 9×8. The depth attribute allows specifying 3D bitmaps, but in this case the depth attribute is set to "1," implying that this is a 2D bitmap. The "tiled" option specifies what should be done if the map references a point outside the map. If the tiled option is set to true, the data is assumed to wrap around continuously.

The main portion of the texture map is the data in between the opening and closing texture tags. The data is encoded row by row, using the Base64 encoding scheme that represents any binary data as text characters. The data used in Figure 5 is a simple checkerboard pattern of 255 and 0.

The `<texmap>` element then associates each of the three vertices of a triangle with a UV coordinate in a 2D texture, or a UVW coordinate in a 3D texture. In Figure 6, the texmap function associates the vertices with a position in the texture map. That resulted in the texture pattern on one of the faces of the tetrahedron. Note that, because the pattern was small, it was "stretched" to fit the face. The "rtexid", "gtexid", "btexid" tell the parser which texture should be used for each one of the red, green and blue color channels. In this case, all three channels are referring to the same texture, resulting in a gray color.

More elaborate 3D textures could be used to specify textures that vary in three dimensions, such as grain pattern of solid wood, or 3D lattices. In fact, the same texturing technique used here to



```
<?xml version="1.0" encoding="utf-8"?>
<amf unit="inch" version="1.1">
  <metadata type="name">Textured Tetrahedron</metadata>
  <object id="1">
    <color><r>1</r><g>1</g><b>1</b></color>
    <mesh>
      <vertices>
        <vertex><coordinates><x>0</x><y>0</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>1</x><y>0.25</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>0.25</x><y>1</y><z>0</z></coordinates></vertex>
        <vertex><coordinates><x>0.4</x><y>0.4</y><z>0.5</z></coordinates></vertex>
      </vertices>
      <volume>
        <triangle><v1>2</v1><v2>1</v2><v3>0</v3></triangle>
        <triangle><v1>0</v1><v2>1</v2><v3>3</v3></triangle>
        <triangle><v1>3</v1><v2>1</v2><v3>2</v3></triangle>
        <triangle><v1>0</v1><v2>3</v2><v3>2</v3>
          <texmap rtexid="1" gtexid="1" btexid="1">
            <u1>0</u1><u2>1</u2><u3>0</u3>
            <v1>0</v1><v2>1</v2><v3>1</v3>
          </texmap>
        </triangle>
      </volume>
    </mesh>
  </object>
  <texture id="1" width="9" height="8" depth="1" type="grayscale" tiled="0">
    /wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A
    /wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A
    /wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A
    /wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A/wD/AP8A
    /wD/AP8A/wD/AP8A/wD/AA==
  </texture>
</amf>
```

Figure 6. A triangle coloring is mapped to a texture. Each of the three vertices of the triangles is mapped to a point in the texture data. The data can be 2D or 3D to allow for either 2D or 3D texture maps. Color images available online at www.liebertpub.com/3dp

pattern color can be used to pattern material composition. But let's leave that for the next tutorial.

References

1. ASTM International. Standard Specification for Additive Manufacturing File Format (AMF) Version 12, ISO/ASTM52915. www.astm.org/Standards/ISOASTM52915.htm (last accessed December 1, 2014).

2. Lipson H. AMF tutorial: The basics (Part 1). *3D Printing and Additive Manufacturing* 2013;1:85–87.

3. AMF wiki. ASTM Additive Manufacturing File Format (AMF). http://STL2.org (last accessed December 1, 2014).

Address correspondence to:
Hod Lipson
242 Upson Hall
Cornell University
Ithaca, NY 14853-7501

E-mail: hod.lipson@cornell.edu