Chem-Expo

CS352: Computer Graphics & Visualization Lab

Project Report

Course Instructor:
Dr. Somnath Dey

Submitted By:

Bhore Parth Shirish	–	200001015
Khushi Verma	–	200001036
Sanskar Verma	–	200001069

Introduction

Chem-Expo explores the possibility of demonstrating fully-animated real-life chemical reactions while integrating them with the fundamental concepts of Computer Graphics and Visualization paradigm. In our project, we have worked with two popular reactions of chemistry, namely – formation of sulfuric acid ($H_2O + SO_3 \rightarrow H_2SO_4$) and bromination of methyl-cyclopentane (methyl-cyclopentane + HBr → bromo-methyl-cyclopentane).

Our project serves two primary purposes. One that it gives a thorough understanding of Computer Graphics course involving experimentation with illumination and shading effects, playing around with various camera/ viewing positions, observing different types of projections – parallel and perspective, image rendering in OpenGL, along with dynamic animation of various components on the screen. Not only this, we learned how to make our project user-friendly and interactive to the user's input. The second place where our project can be extremely useful is in teaching school students the mechanics behind chemical reactions by demonstrating the reactions from our project, which would lend better visualization capabilities for the user.

We started building Chem-Expo from scratch in 2-D with sulfuric acid reaction and rendered simple 2-D molecules in OpenGL with their appropriate coordinates on the screen, along with adding different colors to the atoms and bonds and integrated translation, scaling and rotation via mouse and keyboard. We then proceeded with transforming our 2-D modelling into 3-D modelling and we explored different ways to draw our reactions in 3-D. This was made possible by making spherical and cylindrical shapes for the atoms and bonds respectively. We also had to set the coordinates in 3-D based upon the structure of individual molecules, like tetrahedron for H_2SO_4, planar for SO_3 etc. After having done this, we proceeded with adding the animation part to our project by working out the mathematics and accordingly setting the timer count for the breaking and formation of bonds to clearly demonstrate the actual reaction going on. This was the most crucial as well as thought-provoking task of our project. We built this functionality along with user-interaction so that the reaction could be started/ paused as per the user's whim. We added rotation of individual molecules to the system as well. With this in place, our next task was to add different lighting effects to the reaction and give the user an option to change the camera position. After that, we moved ahead with adding different types of projection schemes to our system like cavalier, cabinet, isometric projections. With everything setup, we added legend, menu, rendered images for each atom on itself, and also prepared an additional reaction of cyclopentane along the same lines.

Specifications

All we need to run the project are the OpenGL library setup and SOIL library included in Ubuntu system.

We can install OpenGL using:

```
$ sudo apt-get update
$ sudo apt-get install libglu1-mesa-dev freeglut3-dev mesa-common-dev
```

To install SOIL library, simply run the command:

```
$ sudo apt-get install libsoil-dev
```

With this done, you can download our code folder. It includes our cpp code files for the chemical reactions we have built with all functionalities included (H2SO4.cpp and cyclopentane.cpp), as well as the sub-folders (soil and img) that are used internally by the files.

Further steps to run the files: (let's take the example of H2SO4.cpp, exactly similar steps and functionalities are there for cyclopentane.cpp, just the name changed)

```
$ g++ H2SO4.cpp -o H2SO4 -lGL -lGLU -lglut -lm -lSOIL
$ ./H2SO4
```

That's it! You will see the reaction window in OpenGL pop-up, with the reaction in place.

Key controls to play around with the reaction:

- Use Keys: W/A (to translate along x-axis), S/D(to translate along y-axis), 5/8(to translate along z-axis).
- Use Mouse: Hold Middle Button and Drag to Zoom In / Zoom Out.
- Use Mouse: Hold Left Button and Drag to Rotate Axis
- Press Enter to Start/Pause animation for the reaction.
- Press Space to Start/Pause rotation of individual molecules about their centers.
- Press P to Change Projection types.
- Press C to Change Camera positions.
- Press L to Change Lighting Effects.
- Press M to Toggle display of Legend.
- Press R to Reset animation from the beginning.
- Press + to Add/Remove display of Axes of the coordinate system.
- Press – to Highlight the colors of molecules.
- Press Esc to Close Window.

Functionalities Implemented

The various functionalities included are:

- **Display of 3-D molecules along with translation/scaling/rotation via user input**: We have used OpenGL's in-built functions like glSphere() and added our own specifications on top of them to get our desired results. We have used Keyboard and Mouse Callback functions to map user input via the keyboard/mouse to corresponding changes in the results using glTranslatef() and glRotatef() functions. We have defined global variables to store the coordinates of each atom/bond and keep updating them as the user wishes. glFlush() and glPostRedisplay() functions help us to re-render/ refresh our screen with each change executed. This also involved mathematical calculations so that we could render the exact geometry of each molecule in 3-D.

  Reference - https://registry.khronos.org/OpenGL-Refpages/gl2.1/xhtml/gluCylinder.xml,

  https://registry.khronos.org/OpenGL-Refpages/gl2.1/xhtml/gluSphere.xml

- **Image and Texture rendering via SOIL library**: We have incorporated SOIL library to render the image of the atom symbol with each atom, for instance an image of 'O' for the oxygen atom, etc. We have set the texture element accordingly to suit the entire reaction.

  Reference - https://bit.ly/3Lr6eqQ

- **Adding Animation to the reaction via Mathematical and Timer concept**: Timing the bonds to be formed and broken at the right time and place was a task that involved various checkpoint timers and mathematical modelling of coordinate system. For example, upto a time t_1, there is no change in the molecules, within t_1 and t_2 we decrease the lengths/radius of bonds that are to break, as they come closer to each other, and within t_2 and t_3, we increase the radius of bonds that are to be newly formed, and finally after t_3, we make the final molecules to translate to their actual positions.

  Reference - https://www.cs.iusb.edu/~danav/teach/c481/c481_07a_anim2d.html

- **Rotation of each molecule w.r.t. its center:** We have implemented rotation of each molecule about its center when the user presses the 'Space' key by first translating the molecule to the origin, performing appropriate rotation, and the translating back to its coordinates. This functionality can be paused/resumed with the 'Space' key too.

- **Integrating various Lighting Effects:** Illuminating concepts taught in class, like ambient light, diffuse light, specular reflection, etc. were helpful to us while building this functionality. We have provided 6 different types of lighting effects to the user, when he presses the 'L' key. First is the default lighting effect of OpenGL in which each object is equally lit from all directions (ambient totally). Specular shine is also reflected on the objects depending on the angle of viewing. Second is an external light source placed in the +z-direction along with ambient light. Third corresponds to the light source being placed in the +z-direction only and no other light source. Fourth similarly refers to light source only in the +x-direction. Fifth is similarly for only +y-direction. Next we have provided an option of random source of light to the user, for now we have set it at an angle from behind, the user can change the code and set the coordinates of his choice. We were able to add all these features with the help of functions like glMaterialfv(), glLightfv() defined in OpenGL and configuring the vectors passed to them as per our needs.

  Reference **-** https://www.glprogramming.com/red/chapter05.html

- **Different Camera positions:** While the user is viewing in perspective projection mode, he can use the 'C' key to switch between various camera positions defined by us. We have used gluLookAt() function to implement this functionality by setting the camera coordinates at different x, y, z positions and getting the view from there.

  Reference **-** http://www.opengl-tutorial.org/beginners-tutorials/tutorial-3-matrices/

- **Toggling among the different Projection types:** This functionality comes out of pure reflection of the concepts taught in class, as we tried to experiment with them ourselves. The key to explore different projection types is ‘P’. By default, what the user sees is perspective projection (from (0,0,2) in the +z-axis), the user can rotate the axis for different angles, change camera positions to see perspective projection from different positions, translate/scale the molecule as well. With the ‘P’ key, the user can transition into Parallel Projection -> Orthographic -> Isometric view, wherein we can see that the direction of projection (DOP) is at equal angles with all the three principal axes. The next in line is Parallel Projection-> Oblique-> Cavalier , wherein the angle between Direction of Projection (DOP) and Plane of Projection (POP) is 45^o. Similarly, next we have Parallel Projection-> Oblique-> Cabinet where the angle is 63.4^o. The user can toggle between these projections. The key technicality behind implementation of these is the use of Matrices for different views and applying multiplication, rotation and appropriate translations on them. For instance- orthographic projection could be simply implemented using the glOrtho() function with the parameters of rotation and angle that we set. For cavalier and cabinet projections, however, we defined our own matrices as studied from other sources and integrated them with other projection functions to get the desired results.

  Reference **-** https://learnopengl.com/Getting-started/Coordinate-Systems

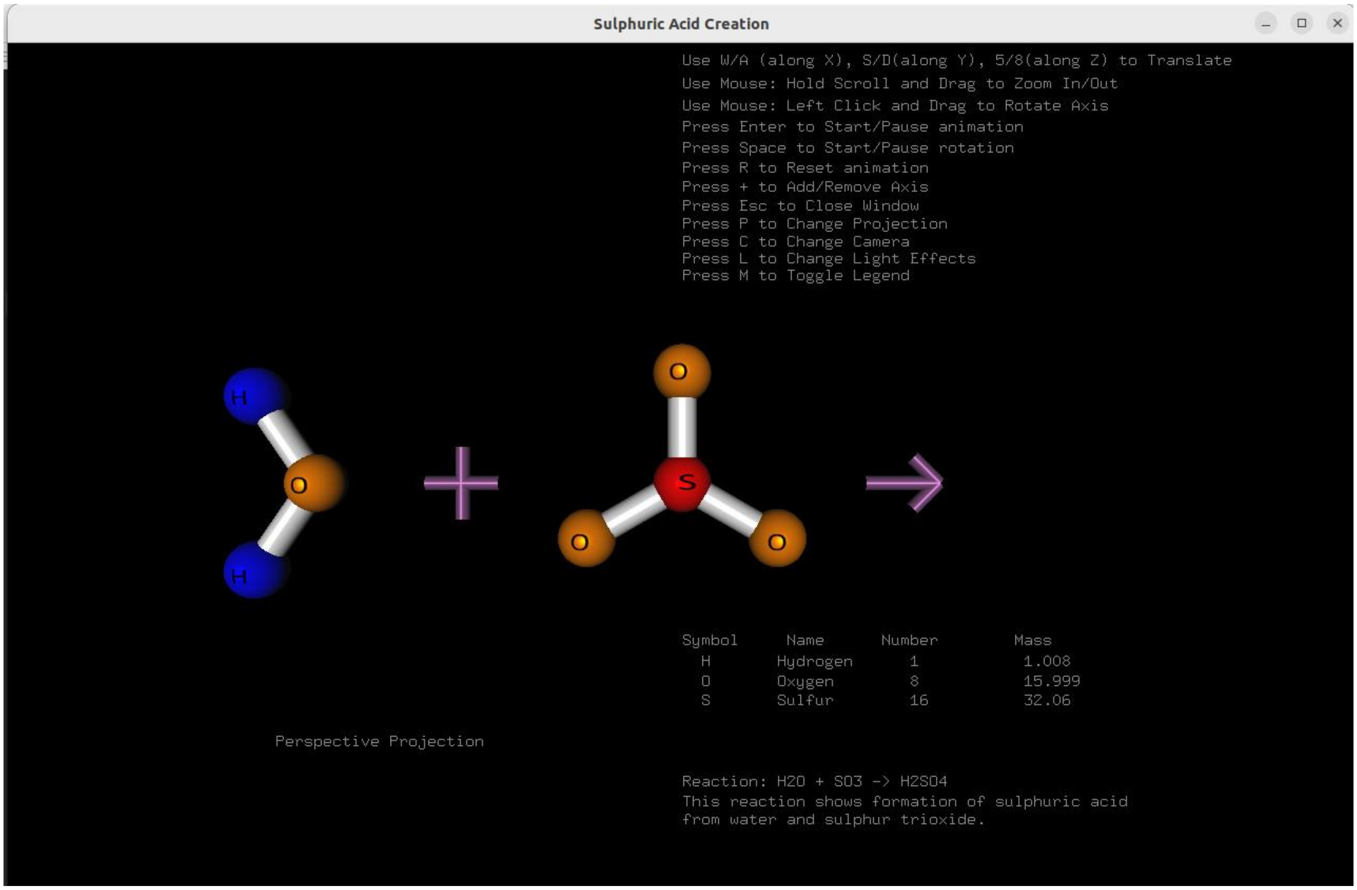
- **Display of Menu/Legend on the Screen along with reaction details:** We used glutBitmapCharacter() function to write the required instructions and basic information about the reaction and display them on the screen. We defined our own coordinates, color and text to be displayed to help the user while experimenting with the reaction. The user can use the ‘M’ key to toggle the display of the legend.

Output

The complete demo of our project has been shown in the video uploaded alongside. It can also be viewed here - https://drive.google.com/file/d/1OsBBYy-6fkQGbtQrVuszhREi3YWLnnlO/view?usp=sharing

Let's dive into the breakdown of our project and have a look at various snippets:

- Original screen for both reactions:

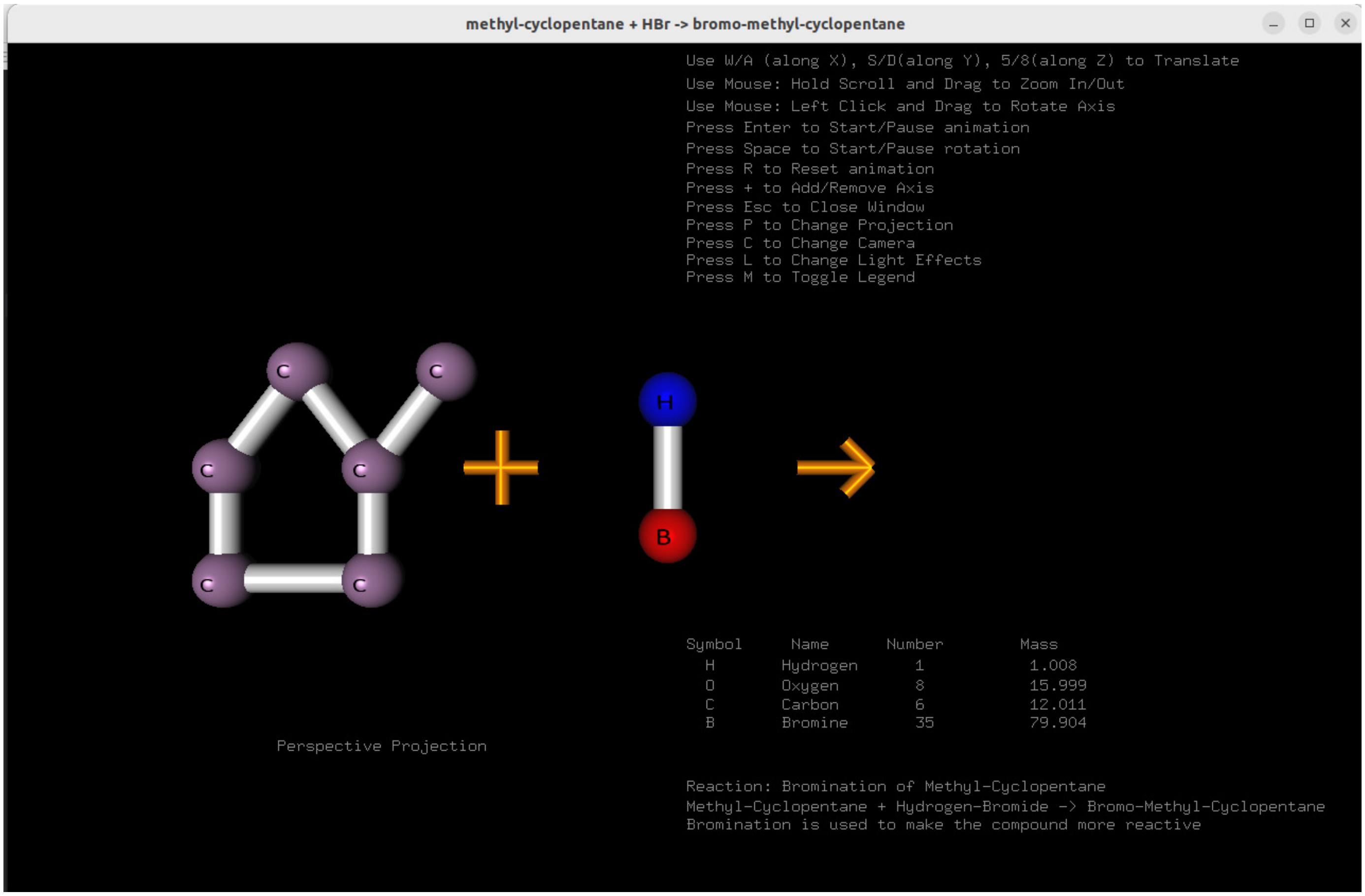


```
Use W/A (along X), S/D(along Y), 5/8(along Z) to Translate
Use Mouse: Hold Scroll and Drag to Zoom In/Out
Use Mouse: Left Click and Drag to Rotate Axis
Press Enter to Start/Pause animation
Press Space to Start/Pause rotation
Press R to Reset animation
Press + to Add/Remove Axis
Press Esc to Close Window
Press P to Change Projection
Press C to Change Camera
Press L to Change Light Effects
Press M to Toggle Legend
```

```
Symbol     Name      Number        Mass
  H       Hydrogen      1            1.008
  O       Oxygen        8            15.999
  S       Sulfur        16           32.06
```

```
Perspective Projection
```

```
Reaction: H2O + SO3 -> H2SO4
This reaction shows formation of sulphuric acid
from water and sulphur trioxide.
```



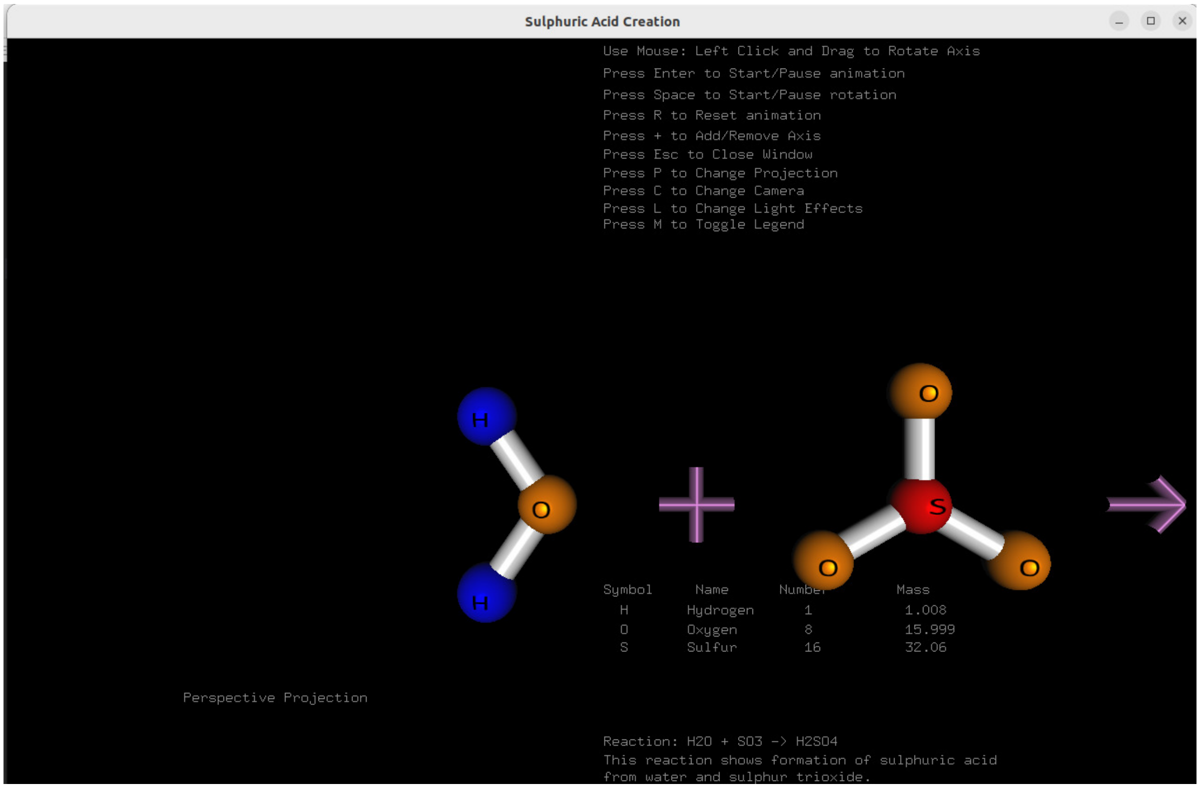
```
Use W/A (along X), S/D(along Y), 5/8(along Z) to Translate
Use Mouse: Hold Scroll and Drag to Zoom In/Out
Use Mouse: Left Click and Drag to Rotate Axis
Press Enter to Start/Pause animation
Press Space to Start/Pause rotation
Press R to Reset animation
Press + to Add/Remove Axis
Press Esc to Close Window
Press P to Change Projection
Press C to Change Camera
Press L to Change Light Effects
Press M to Toggle Legend
```

```
Symbol     Name       Number        Mass
  H       Hydrogen      1            1.008
  O       Oxygen        8            15.999
  C       Carbon        6            12.011
  B       Bromine       35           79.904
```

Perspective Projection

```
Reaction: Bromination of Methyl-Cyclopentane
Methyl-Cyclopentane + Hydrogen-Bromide -> Bromo-Methyl-Cyclopentane
Bromination is used to make the compound more reactive
```

- After some translations/ scaling:



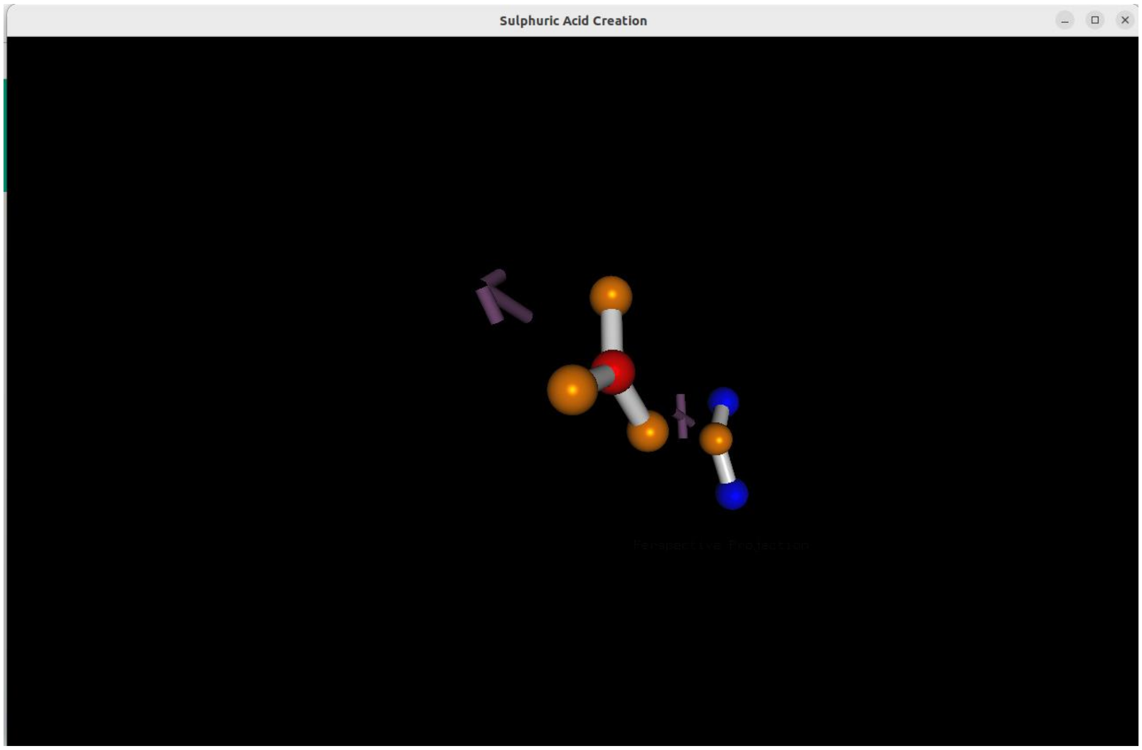
```
Use Mouse: Left Click and Drag to Rotate Axis
Press Enter to Start/Pause animation
Press Space to Start/Pause rotation
Press R to Reset animation
Press + to Add/Remove Axis
Press Esc to Close Window
Press P to Change Projection
Press C to Change Camera
Press L to Change Light Effects
Press M to Toggle Legend
```

```
Symbol     Name      Number       Mass
  H       Hydrogen     1           1.008
  O       Oxygen       8           15.999
  S       Sulfur       16          32.06
```

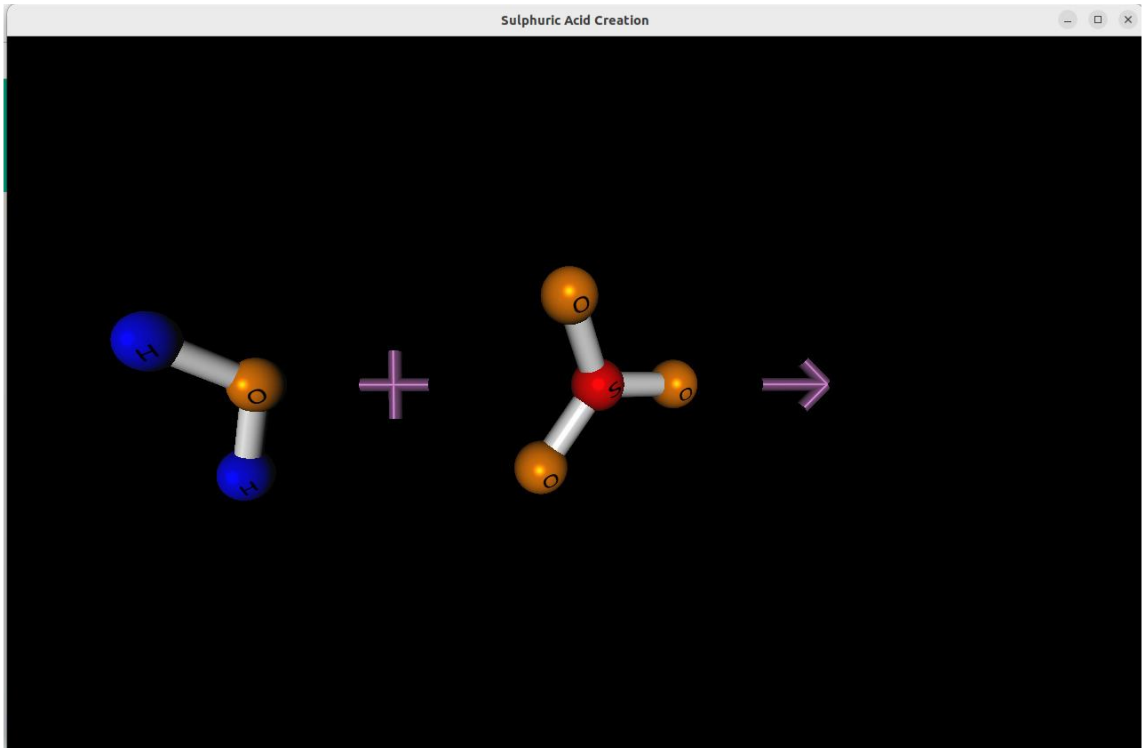
Perspective Projection

```
Reaction: H2O + SO3 -> H2SO4
This reaction shows formation of sulphuric acid
from water and sulphur trioxide.
```

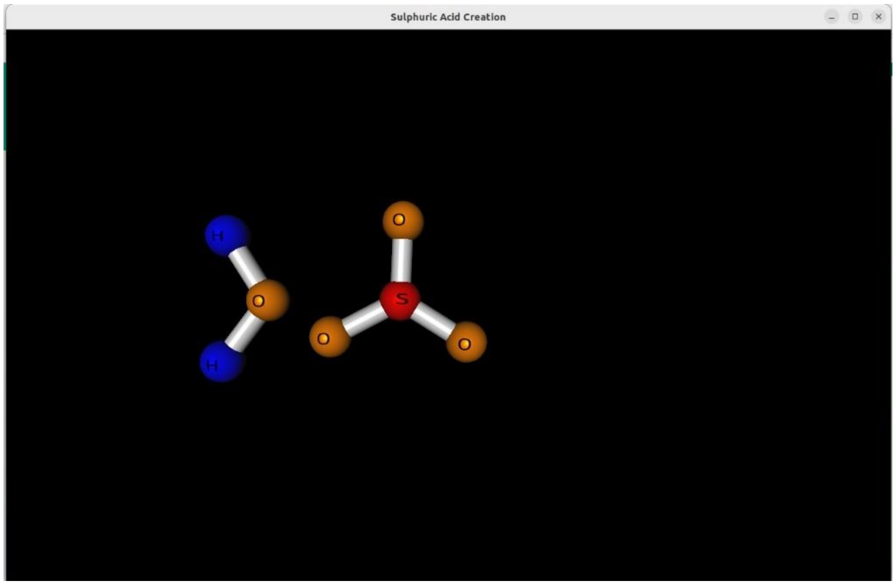
- Let’s toggle the menu, to no display and perform some rotation to the axes:

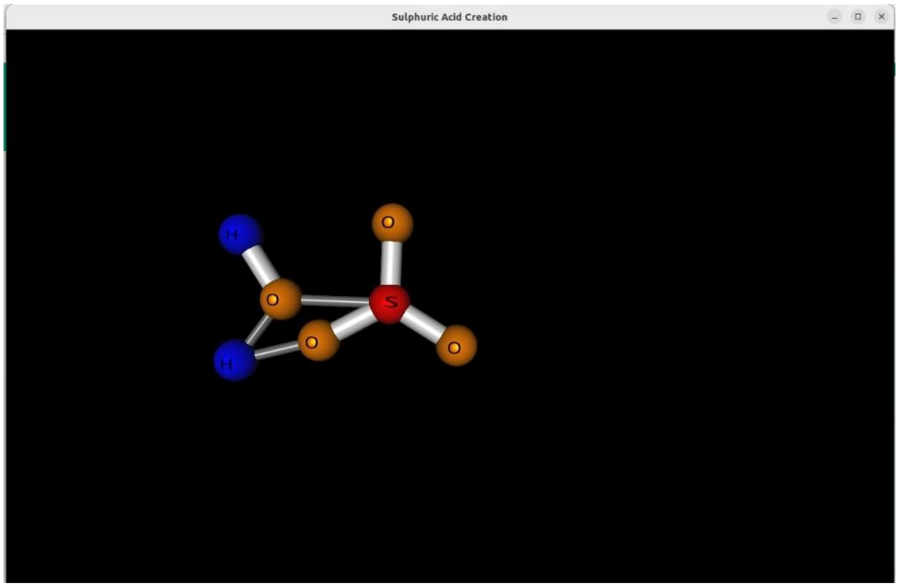


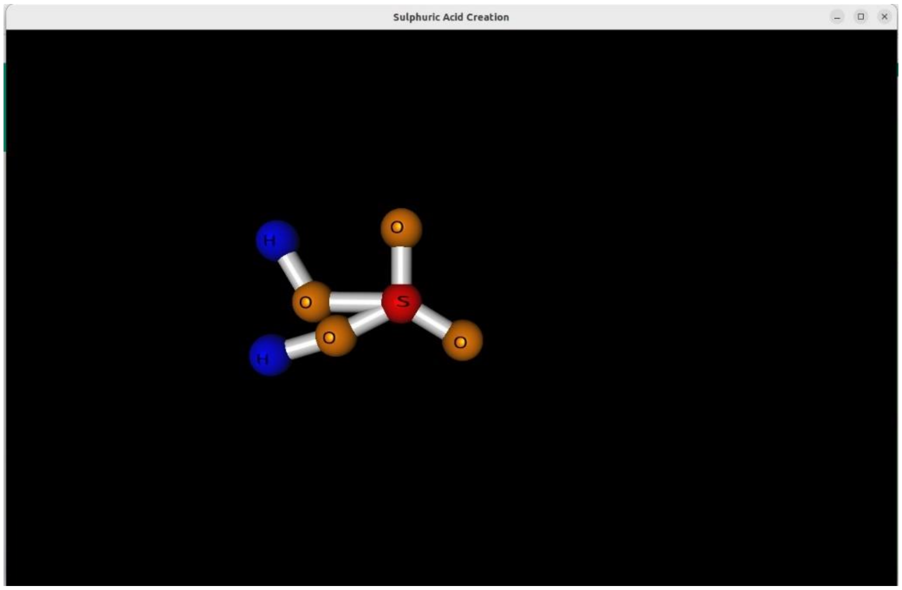
- Now, let’s start the rotation of each molecule with ‘Space’ key:



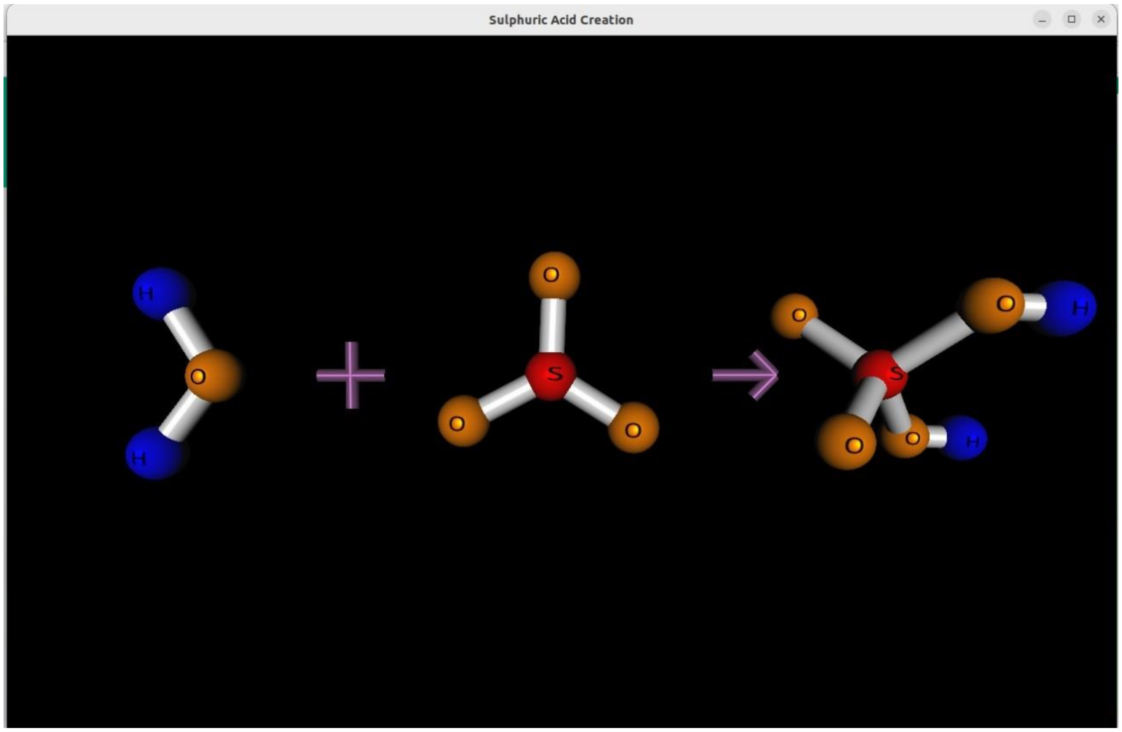
- We can go ahead and start the animation of reaction, the various stages in it are as:



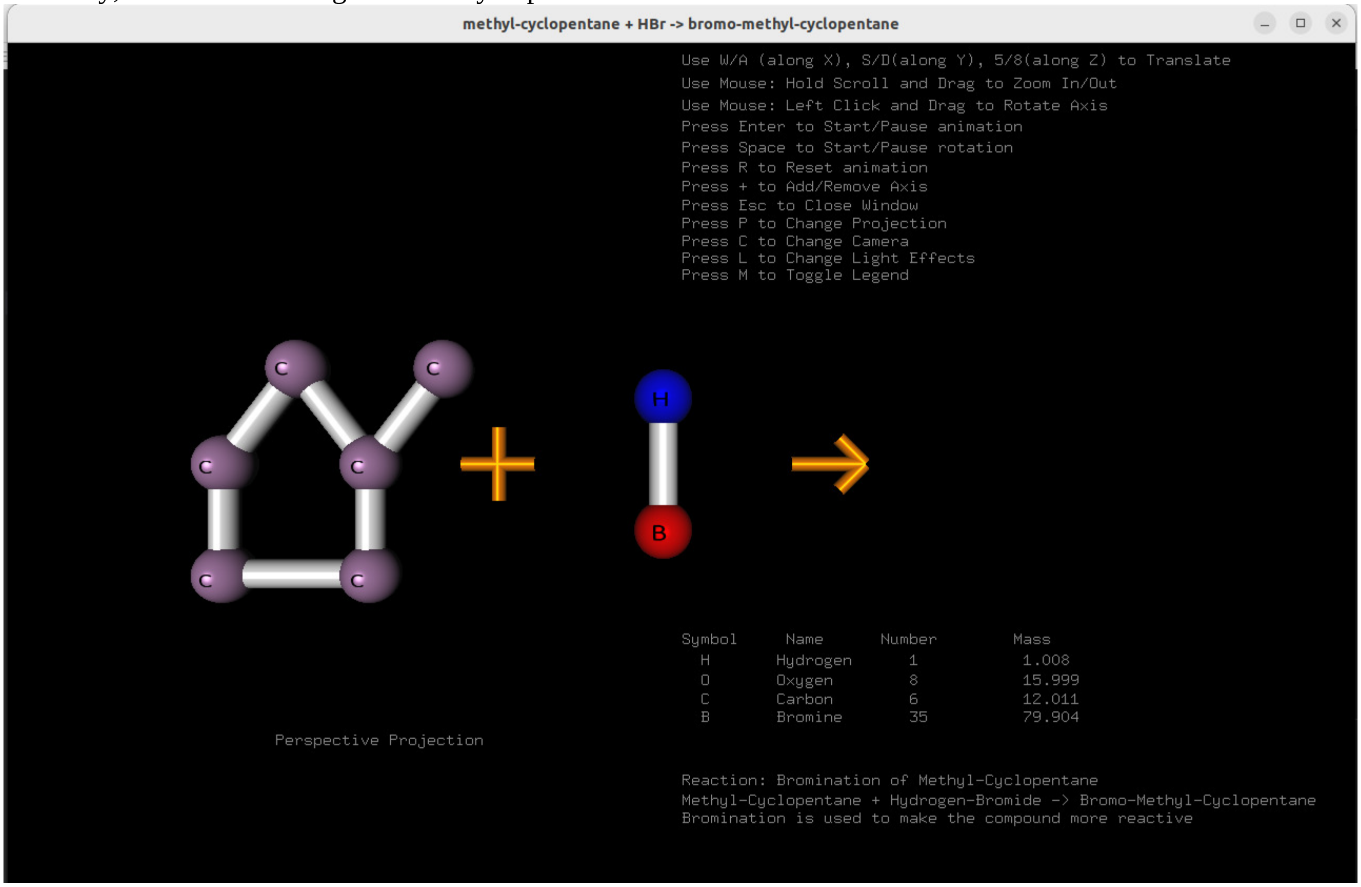




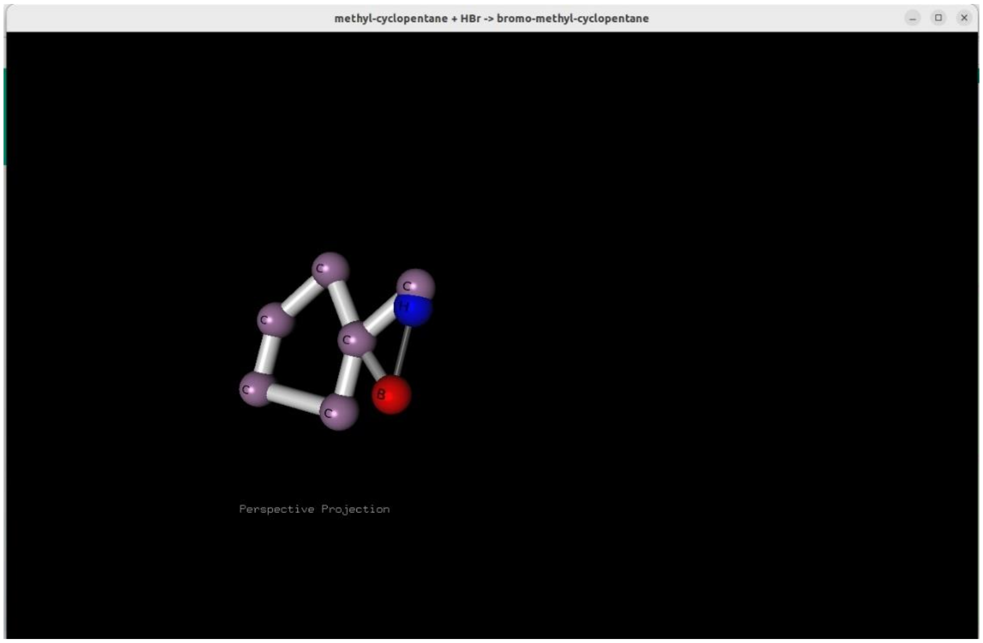
Finally, we get our complete reaction:



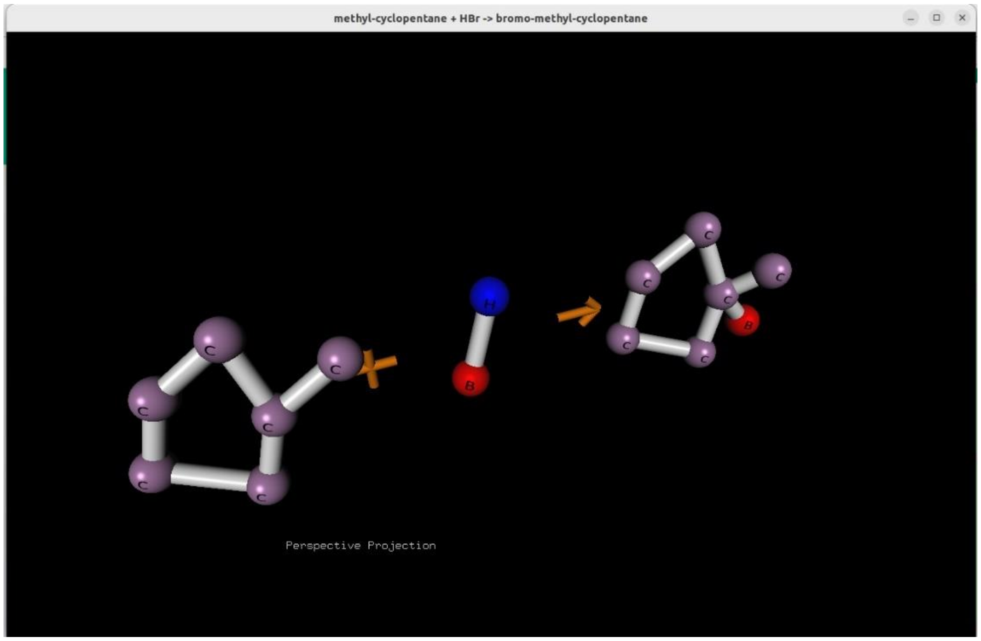
Similarly, we have these stages for the cyclopentane reaction:



```
Use W/A (along X), S/D(along Y), 5/8(along Z) to Translate
Use Mouse: Hold Scroll and Drag to Zoom In/Out
Use Mouse: Left Click and Drag to Rotate Axis
Press Enter to Start/Pause animation
Press Space to Start/Pause rotation
Press R to Reset animation
Press + to Add/Remove Axis
Press Esc to Close Window
Press P to Change Projection
Press C to Change Camera
Press L to Change Light Effects
Press M to Toggle Legend

Symbol     Name       Number       Mass
  H      Hydrogen      1           1.008
  O      Oxygen        8           15.999
  C      Carbon        6           12.011
  B      Bromine       35          79.904

Perspective Projection

Reaction: Bromination of Methyl-Cyclopentane
Methyl-Cyclopentane + Hydrogen-Bromide -> Bromo-Methyl-Cyclopentane
Bromination is used to make the compound more reactive
```



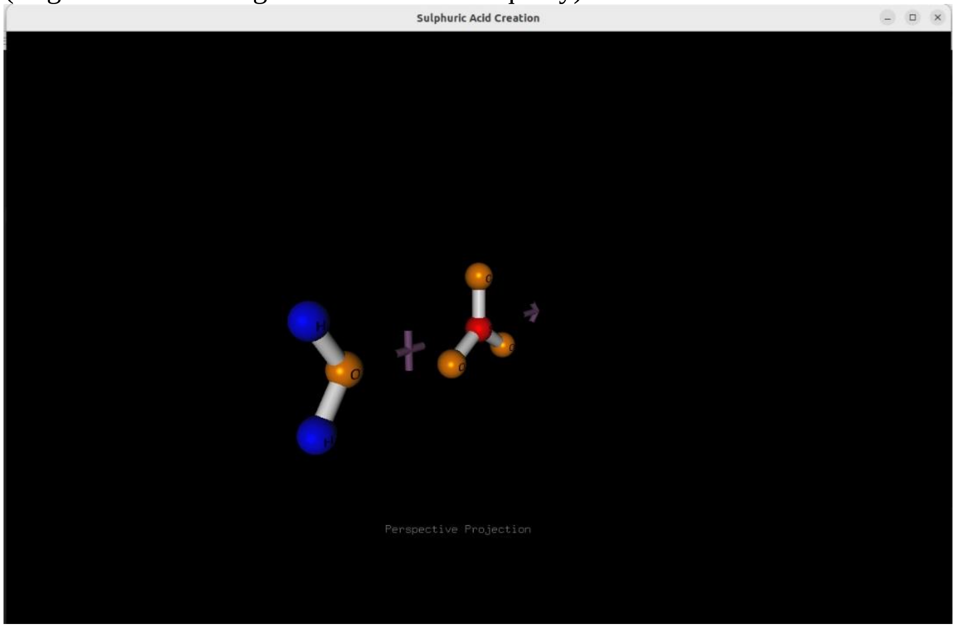
methyl-cyclopentane + HBr -> bromo-methyl-cyclopentane



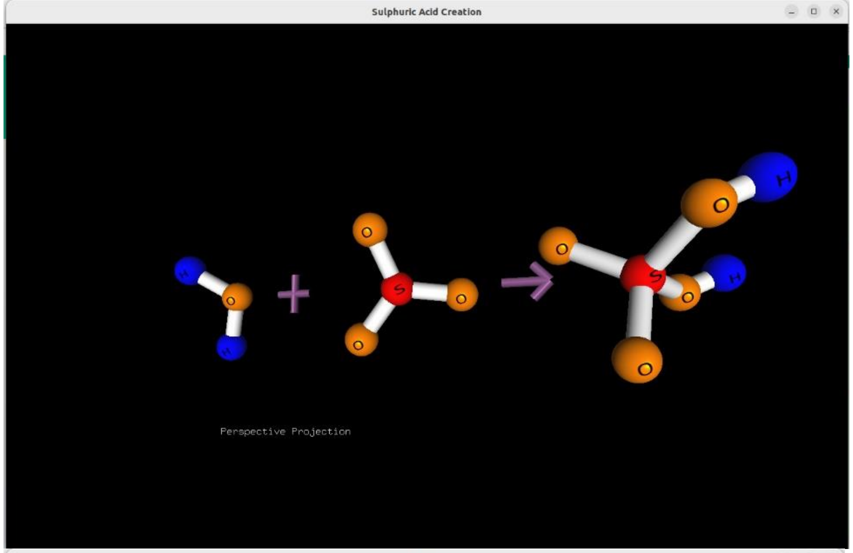
methyl-cyclopentane + HBr -> bromo-methyl-cyclopentane

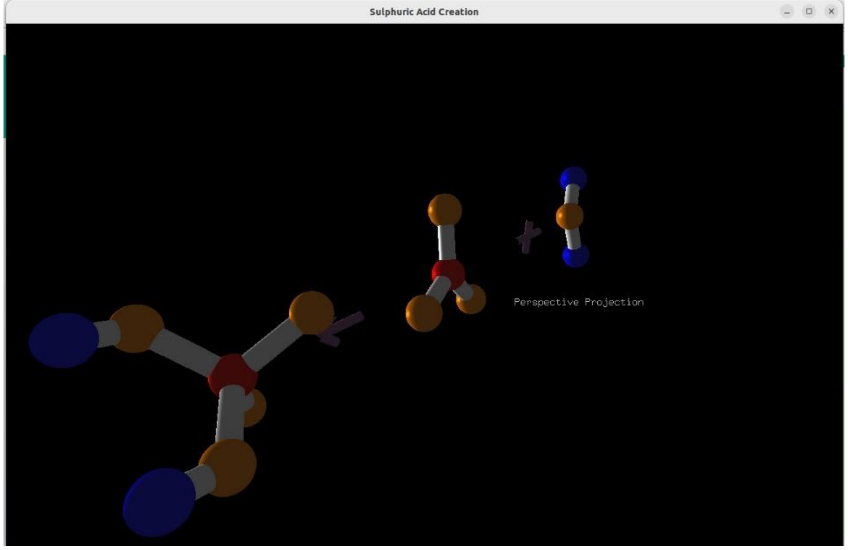
- Next, let's dive into the different lighting effects we have:

(original – ambient light in all directions equally):

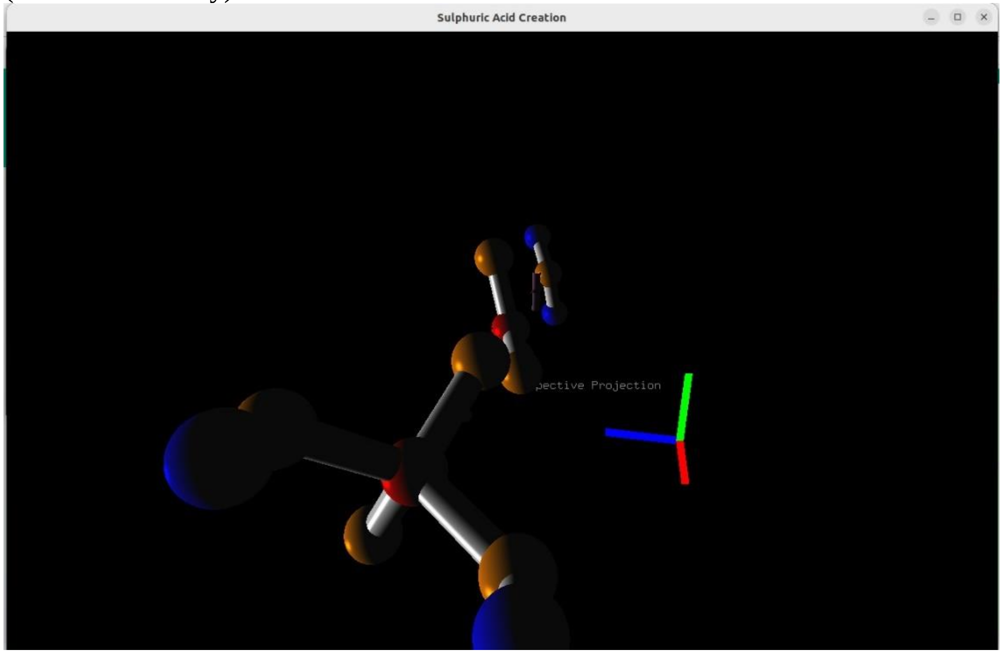


(ambient + from +z-axis):

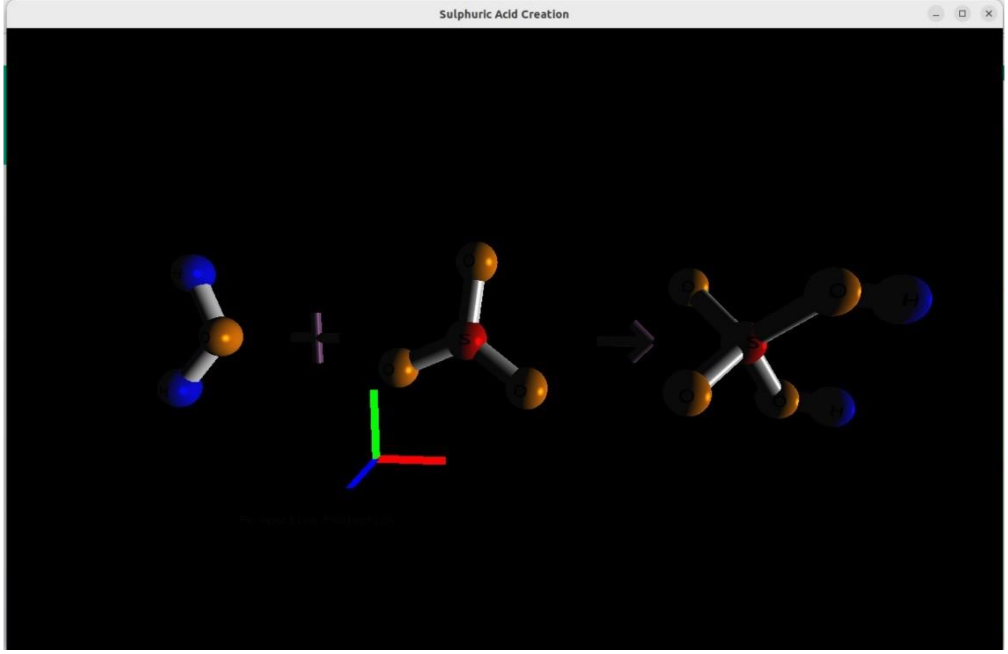




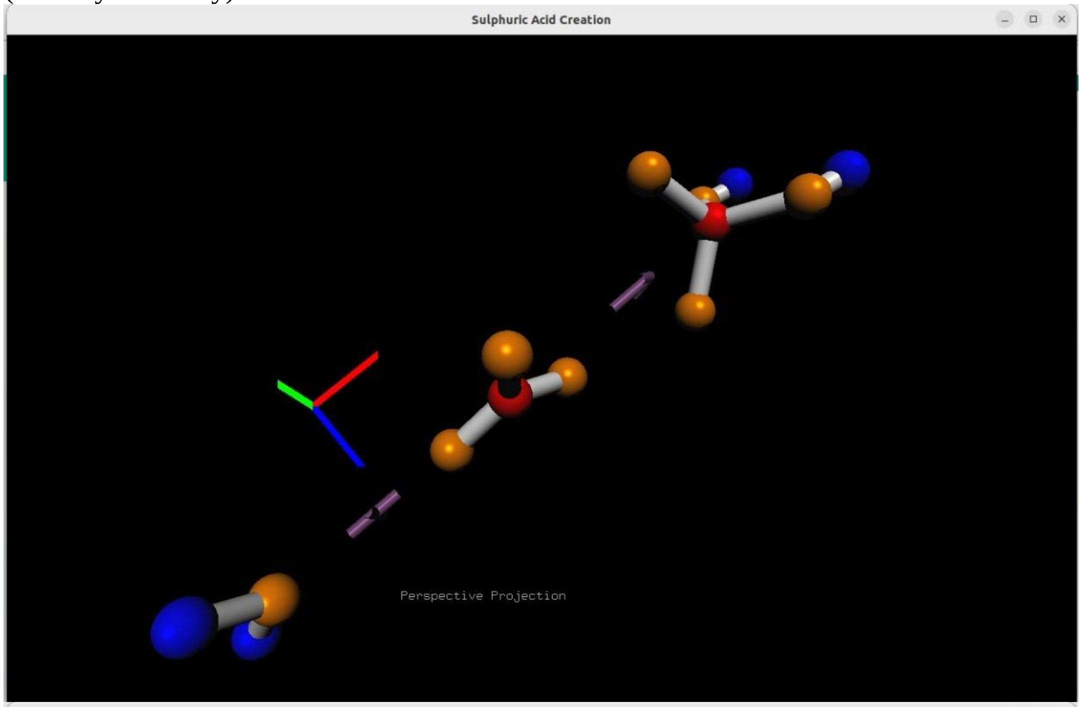
(from +z-axis only):

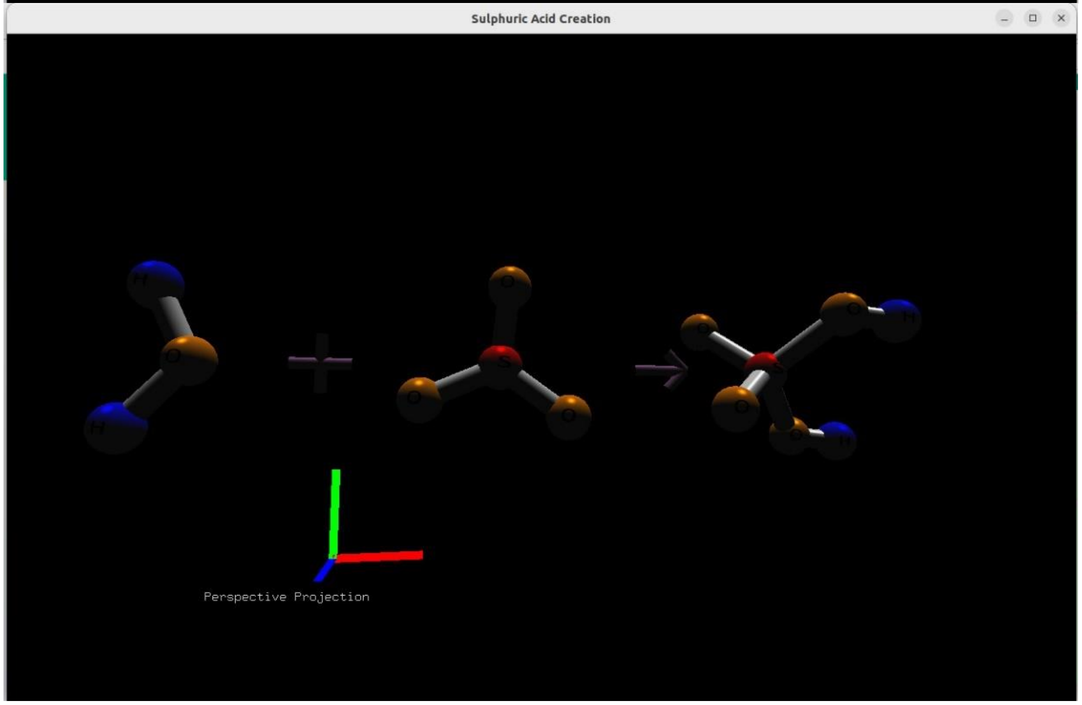


(from +x-axis only):

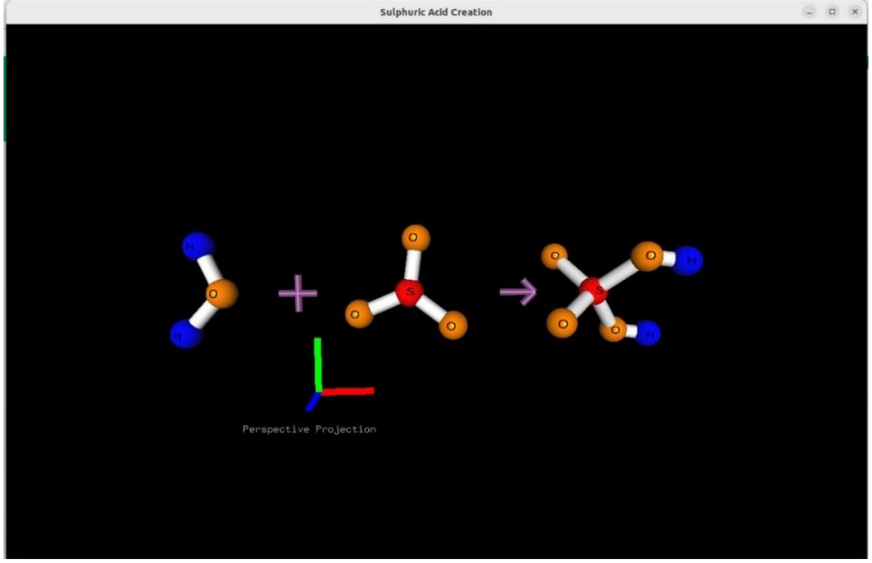


(from +y-axis only):

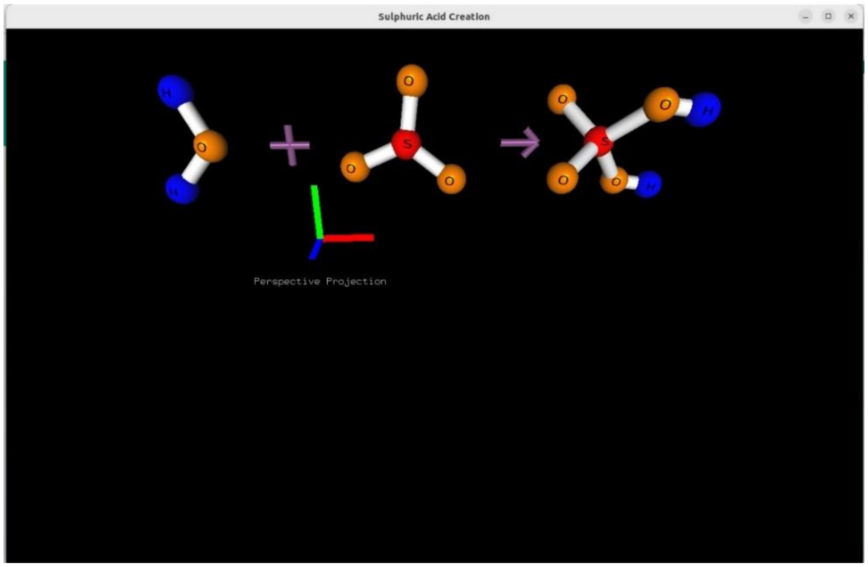




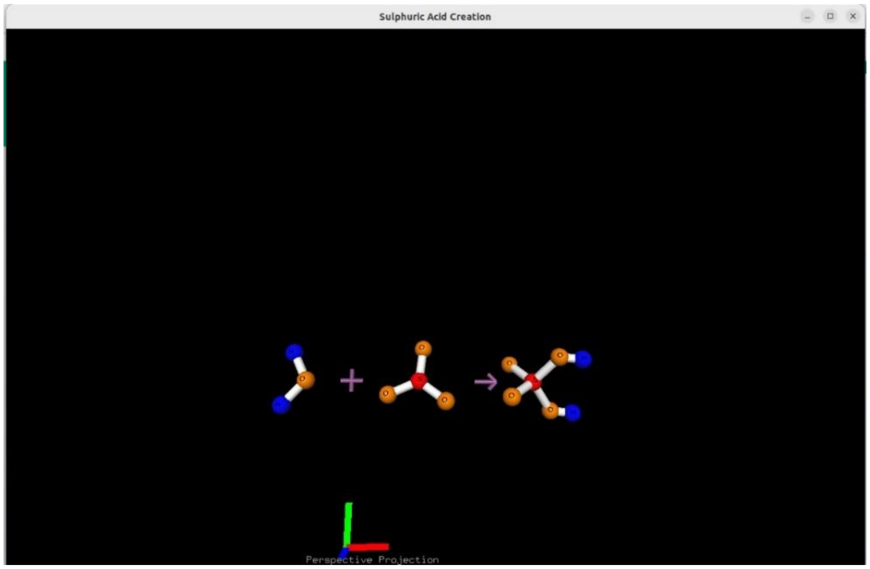
- Different Camera Positions in Perspective Projection:



(from (0,0,2))



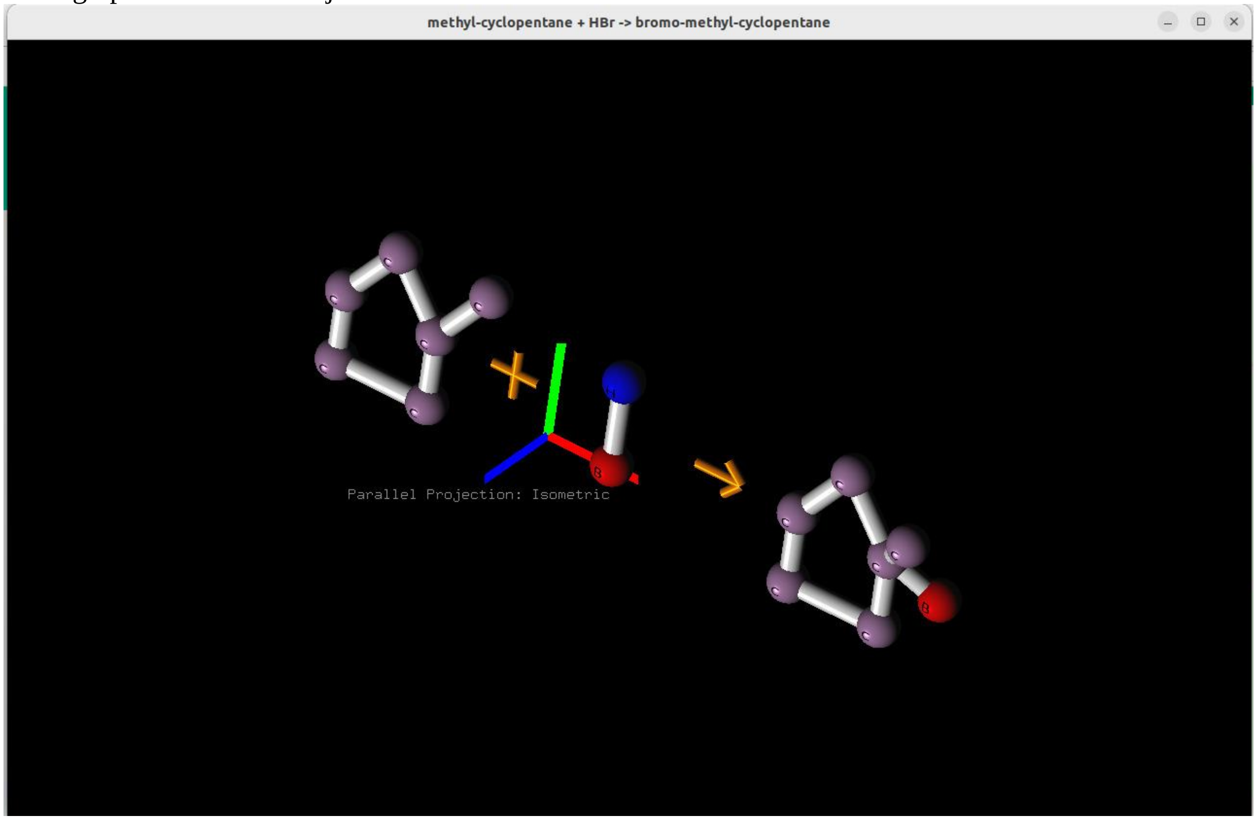
(from (0,0.8,2))



(from (0,-0.8,2))

- Other Parallel Projections using ‘P’ key:

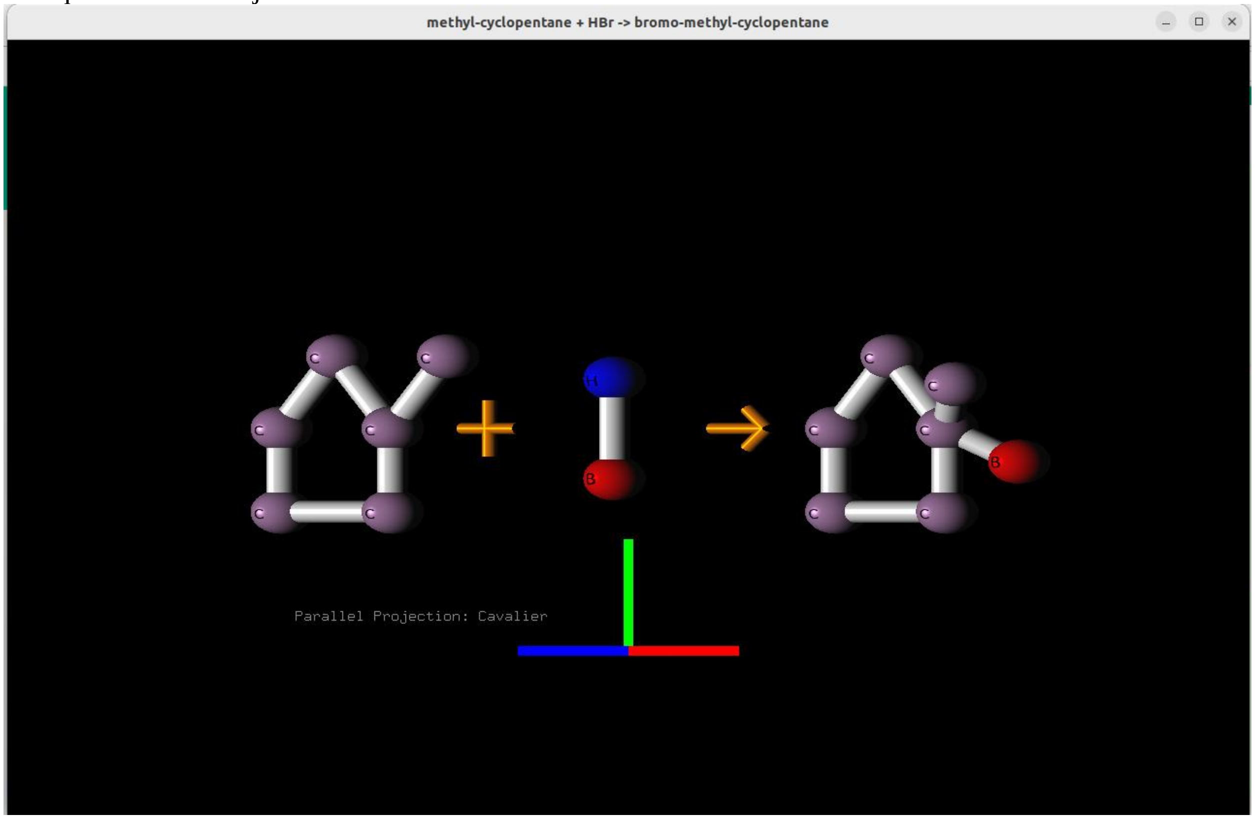
Orthographic Isometric Projection:



methyl-cyclopentane + HBr -> bromo-methyl-cyclopentane

Parallel Projection: Isometric

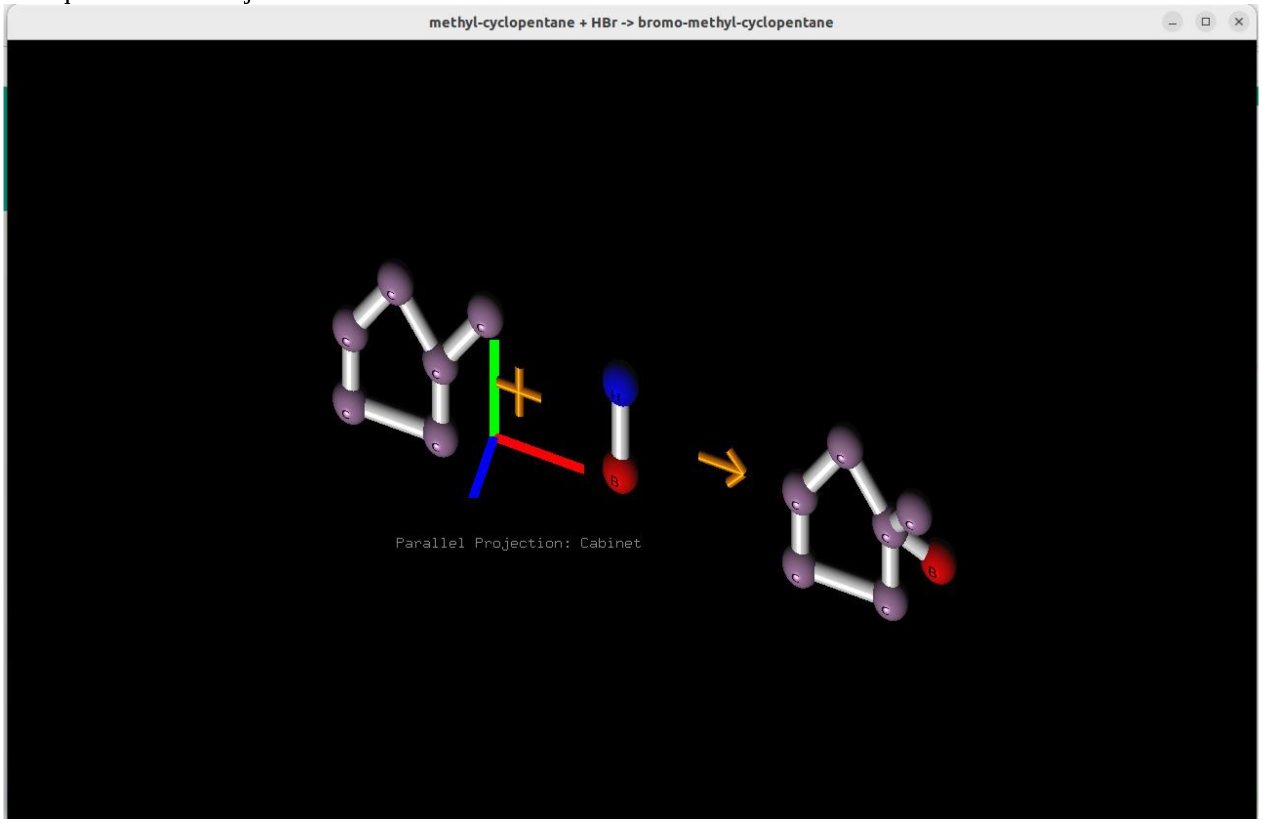
Oblique Cavalier Projection:



methyl-cyclopentane + HBr -> bromo-methyl-cyclopentane

Parallel Projection: Cavalier

Oblique Cabinet Projection:



methyl-cyclopentane + HBr -> bromo-methyl-cyclopentane

Parallel Projection: Cabinet

That was about our project. Thank you!