MATLAB Answers™

Vote

0

# Why is my surf animation so slow?

Asked by Phil on 9 Apr 2014
**Latest activity** Commented on by Phil on 9 Apr 2014
Accepted Answer by Sven
34 views (last 30 days)

I have just created the basis for my project and I'm almost happy with it apart from the speed at which it runs. My laptop definitely shouldn't be struggling with this. It seems to draw every 1 second or so which is no where near quick enough. I have tried using get frame and others but its still the same.

The code is below.

clc; clear all;

[X,Y]=meshgrid(-18*pi:.1:18*pi,-18*pi:.1:18*pi);

a=1; b=0.001; R=(X.^2+Y.^2);

while 1

```
a=a-0.05;
```

```
Z=(sin(sqrt(R*a))./exp(R*b));
```

```
s=surf(X,Y,Z);
    colormap(cool) ;
    set(s, 'edgecolor','none');
    alpha(0.9);
    view(45,60);
```

```
    xlim([-18*pi, 18*pi]); ylim([-18*pi, 18*pi]);  % static limits
```

```
pause(0.1)
```

end

## 0 Comments

Log in to comment.

## Tags

surf speed pause getframe slow project

## Products

*No products are associated with this question.*

## Related Content

## 1 Answer

Vote

0

Link

Answer by Sven on 9 Apr 2014

Edited by Sven on 9 Apr 2014

✓ Accepted Answer

Hi Phil,

There are a few things you can do to speed this up.

Firstly, the **pause** command will force your loop to sit idle for 0.1 seconds each loop, so you will have at *most* 10 frames a second. Replace it with **drawnow** and you'll see a speedup.

Next, there are a few things that are called within the loop that are basically static and unchanging. These can be brought out of the loop (ie, called only once) and doing so will save some time within it:

```
        alpha(0.9);
        view(45,60);
        xlim([-18*pi, 18*pi]); ylim([-18*pi, 18*pi]);  % static limits
```

But, the biggest culprit is that you're calling **surf()** on *every iteration* of your loop. Basically, this means that you're adding a new plot object every iteration, rather than just updating the contents of the old one. This will slow things down almost immediately as MATLAB isn't that great at collecting a large number of complex display objects.

Here's my take on the loop that you had (I'm just looping 10 times rather than forever):

```
figure
[X,Y]=meshgrid(-18*pi:.1:18*pi,-18*pi:.1:18*pi);
s = surf(X,Y,X*0); % Get a handle to a single graphics object
% Set up the view once
colormap(cool) ;
set(s, 'edgecolor','none');
% alpha(0.9);
view(45,60);
xlim([-18*pi, 18*pi]); ylim([-18*pi, 18*pi]);  % static limits
a=1; b=0.001; R=(X.^2+Y.^2);
```

```
for jjj = 1:10
    a=a-0.05;
    Z=(sin(sqrt(R*a))./exp(R*b));
    set(s, 'ZData',Z);
    drawnow
end
```

It runs at around 2.5 frames per second. Almost all of this is in the display portion (updating the surface location)... basically MATLAB is very fast on the calculations but 3d plotting speed isn't one of its strengths.

You'll also notice that I've removed the **alpha** call. This has a huge impact on speed, as a different (more complex) rendering algorithm is needed if there are transparent objects in the view. Without the alpha call (which didn't *seem* to change the end result very much), things are a lot faster.

Hope that answered your question. If you really need a particular frame rate, I can suggest reducing the size of your X and Y matrices. Depending on how large you'll have the figure on the screen, having X and Y matrices of one third the size (-18*pi:.3:18*pi) increased the frame rate to around 16fps on my machine and my eye couldn't tell the difference in the rendered output.

Thanks, Sven.

## 1 Comment

Phil on 9 Apr 2014

Thanks Sven, I've played around with the code you wrote, personalised it a bit so it makes sense to me and its running much quicker. It seems that the transparency was the killer as if I ever try to add it back in the frame rate drops dramatically.

Thanks again.