How long is my SVG <text> element?

David J. Birnbaum and Charlie Taylor, University of Pittsburgh

Presented at Balisage 2021

The Problem

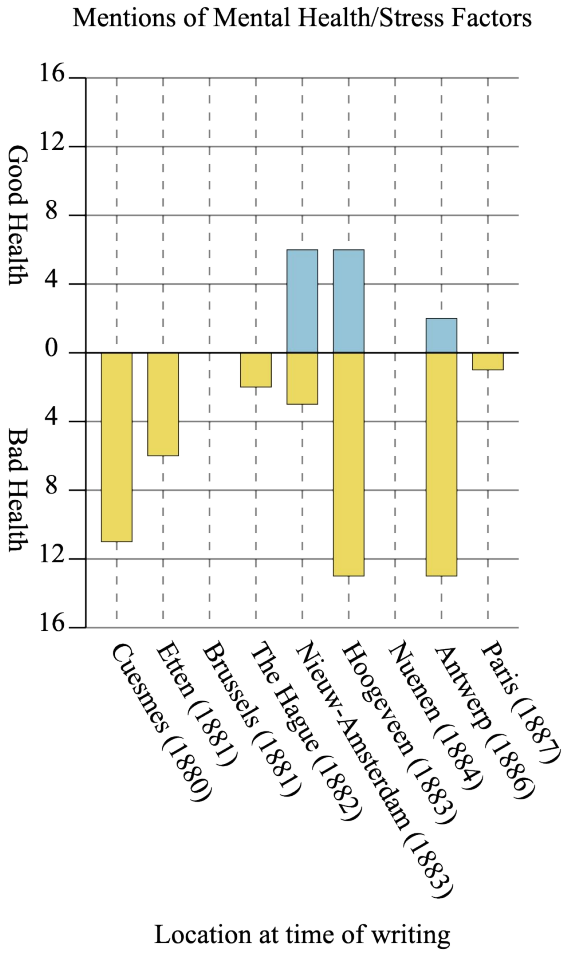
Developers rely on dimensions of objects to determine their position

- Text rendering depends on the widths of individual characters — information is not accessible in a regular developing environment

Problem example: Van Gogh data

Which <location> element will take up the most space when transformed to an SVG <text> element?

How long will that <text> element be?



XSLT approach

Query font metrics (extracted from TrueType TTF files and formatted as XML), create mapping from characters to their given width

Dimensions of **<text>** bounding box will be

1. Height: the font size (approximately)
2. Width: the sum of character widths in the string

E.g., width of “A” in Times New Roman, size 16 is 11.5546875 px.

XSLT Approach: font metrics formatted as XML

```
<character dec="65" hex="0x41" name="A" str="A" width="11.456"/>
<character dec="66" hex="0x42" name="B" str="B" width="10.672"/>
<character dec="67" hex="0x43" name="C" str="C" width="10.672"/>
<character dec="68" hex="0x44" name="D" str="D" width="11.552"/>
<character dec="69" hex="0x45" name="E" str="E" width="9.776"/>
<character dec="70" hex="0x46" name="F" str="F" width="8.896"/>
<character dec="71" hex="0x47" name="G" str="G" width="11.552"/>
<character dec="72" hex="0x48" name="H" str="H" width="11.552"/>
<character dec="73" hex="0x49" name="I" str="I" width="5.328"/>
```

XSLT approach

Length: 67.9921875
Test string

Length: 69.328125
AVAVAV

Length: 0

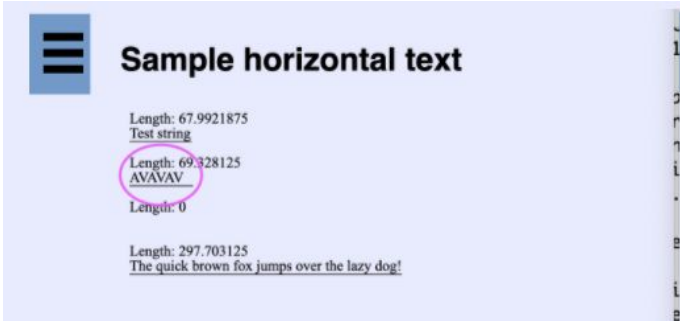
Length: 297.703125
The quick brown fox jumps over the lazy dog!

Test string

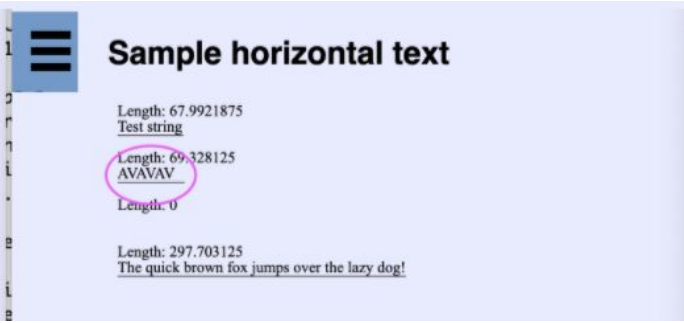
AVAVAV

The quick brown fox jumps over the lazy dog!

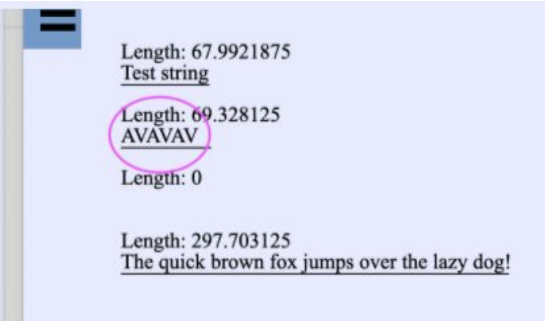
XSLT approach



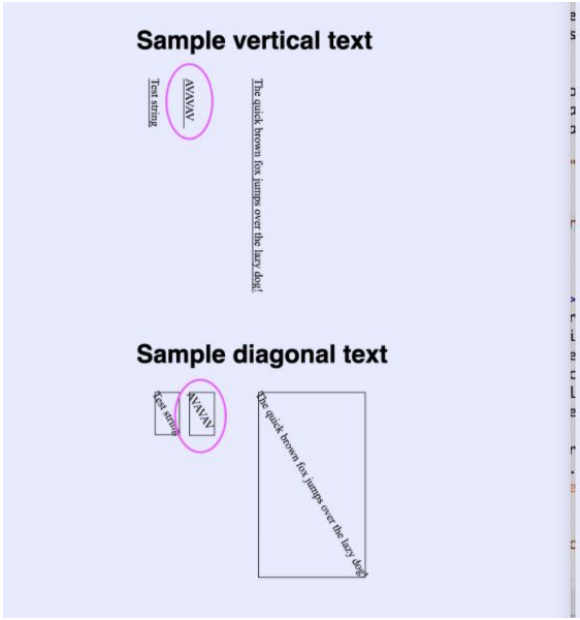
Firefox



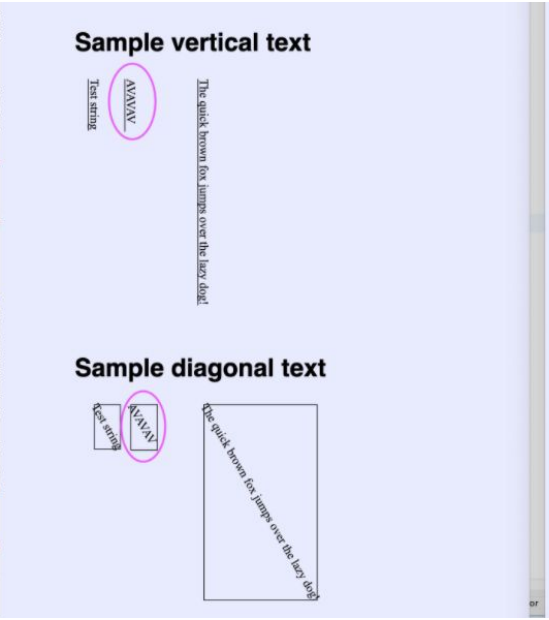
Chrome



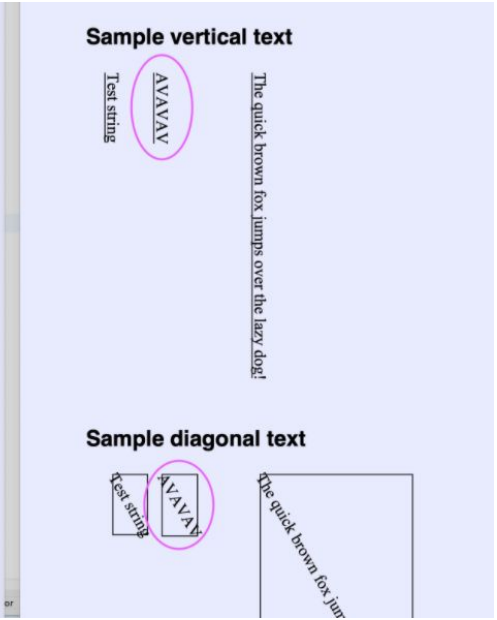
Safari



Firefox



Chrome

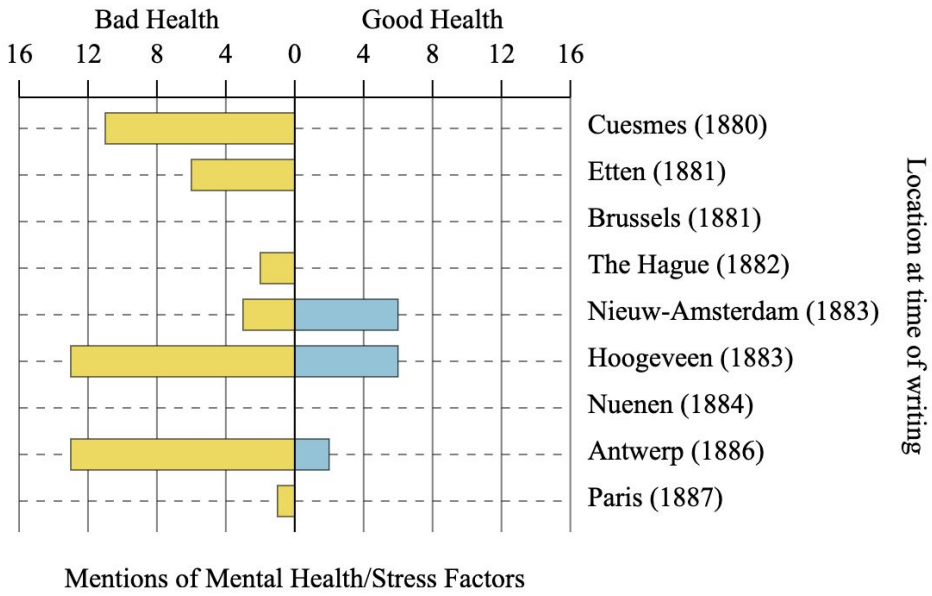


Safari

XSLT approach

Calculate sum of character widths and find the longest

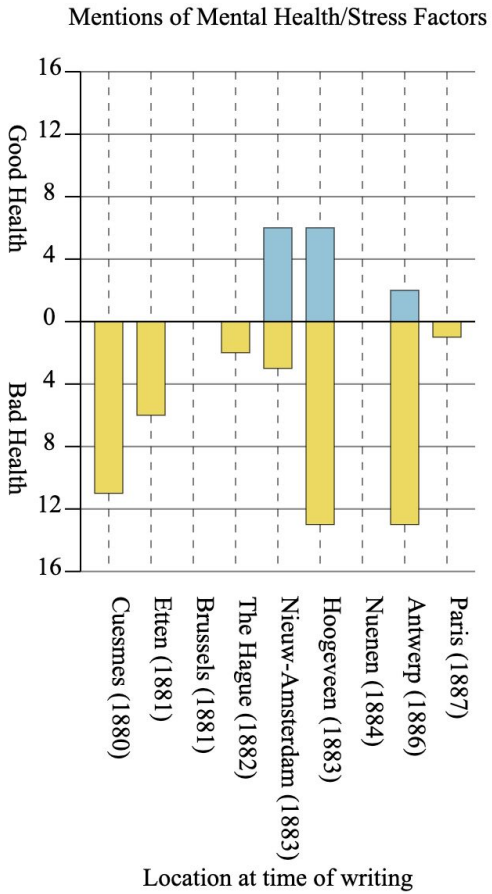
Treat this as a dimension of bounding box



XSLT approach

Calculate sum of character widths and find the longest

Treat this as a dimension of bounding box

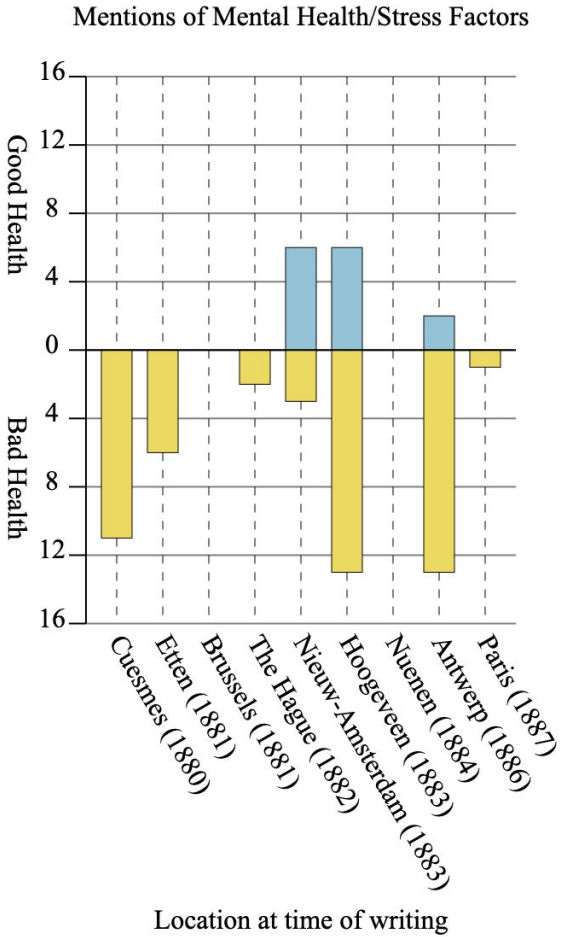


XSLT approach: diagonal text

Sum of character widths represents length of diagonal text

Use this alongside the known angle of rotation to find the **vertical** space occupied by text

E.g., *hypotenuse * cosθ*



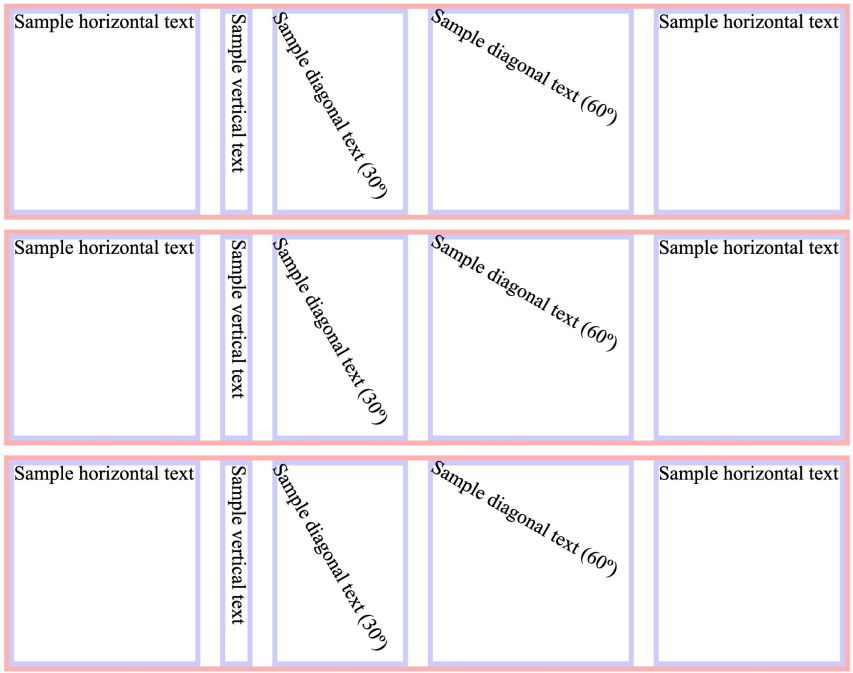
JavaScript approach

Output components as separate **<svg>** elements, **@height** and **@width** not specified

Compute dimensions of bounding box using the **element.getBBox()** function

Write those dimensions into the DOM, arrange components using CSS Flexbox

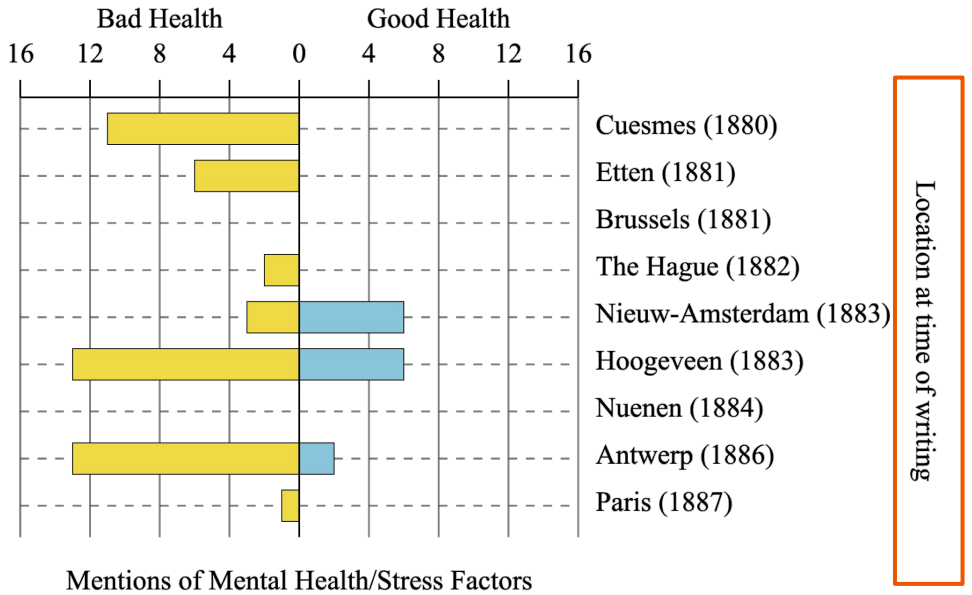
JavaScript approach



JavaScript approach

Graph divided into multiple **<svg>** elements wrapped in HTML ** ** elements

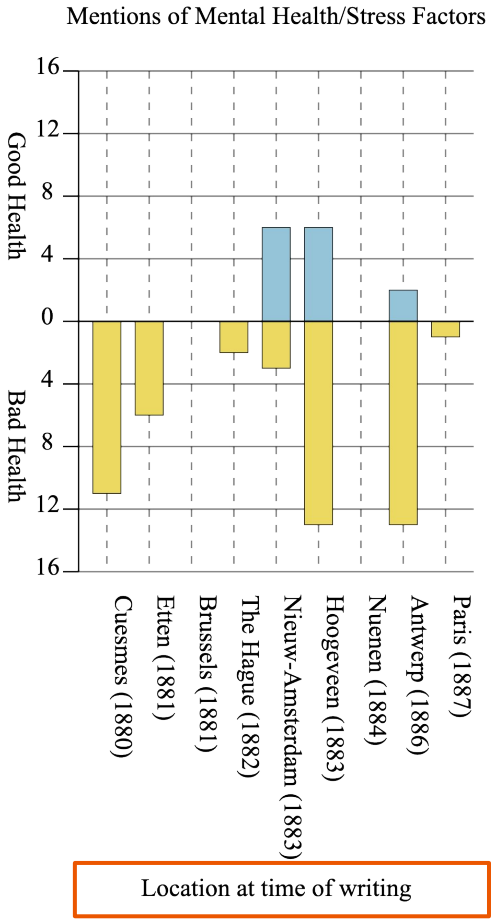
Arrange ** ** elements using CSS Flexbox



JavaScript approach

Graph divided into multiple **<svg>** elements wrapped in HTML ** ** elements

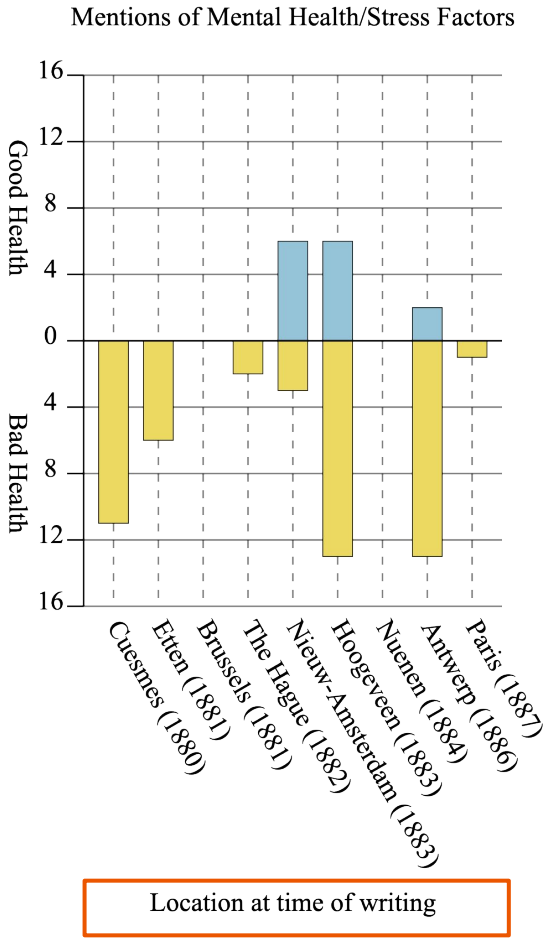
Arrange ** ** elements using CSS Flexbox



JavaScript approach

Graph divided into multiple **<svg>** elements wrapped in HTML ** ** elements

Arrange ** ** elements using CSS Flexbox



Centering text in ellipses

Using XSLT approach, compute **@rx** (horizontal radius) as half the length of the text, plus padding

@cx and **@cy** values of ellipses will equal **@x** and **@y** values of text

(when **@dominant-baseline** and **@text-anchor** are **"middle"**)

XSLT approach

+ Typical of the way we usually work with SVG
+ Generally not browser-dependant (except for browser kerning)
+ Easier to use outside browser environments

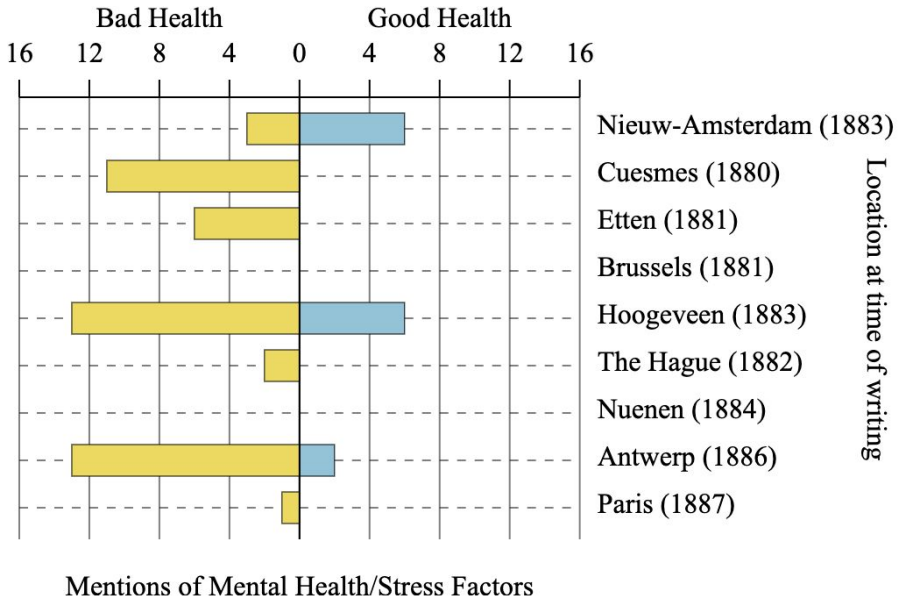
- Only accesses glyph widths, ignores font effects
- Requires preparation of metrics for every font family/size used

JavaScript approach

+ No preparation of font metric info
+ Adapts to viewing environment: dimensions will be correct no matter the font
+ Does not require dealing with font effects

- Requires knowledge of JavaScript
- Cannot create SVG independent of an HTML file

Loose ends


Bad Health
Good Health
16
12
8
4
0
4
8
12
16
Nieuw-Amsterdam (1883)
Cuesmes (1880)
Etten (1881)
Brussels (1881)
Hoogeveen (1883)
The Hague (1882)
Nuenen (1884)
Antwerp (1886)
Paris (1887)
Location at time of writing
Mentions of Mental Health/Stress Factors

Questions and comments?

David J. Birnbaum
Professor of Slavic Languages and Literatures
University of Pittsburgh

djbpitt@gmail.com

Charlie Taylor
Undergraduate teaching assistant
University of Pittsburgh

CLT76@pitt.edu