CUDASCII

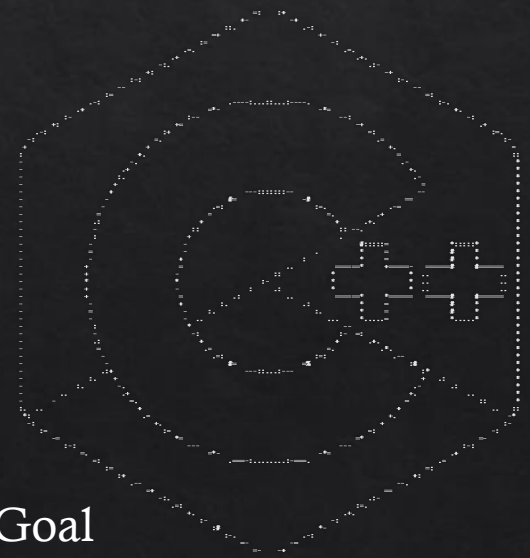
A CUDA-accelerated image-to-ascii Python library written in modern C++

Presented by Alex Groeger & Alex Schwartz

The Goal

- Convert an image into text
  - Must be fast enough to process a 1080p 60 fps video stream
  - Text best preserves the original resolution when viewed in a terminal
  - Characters best match the underlying pixel intensities instead of average intensity

Initial Goal



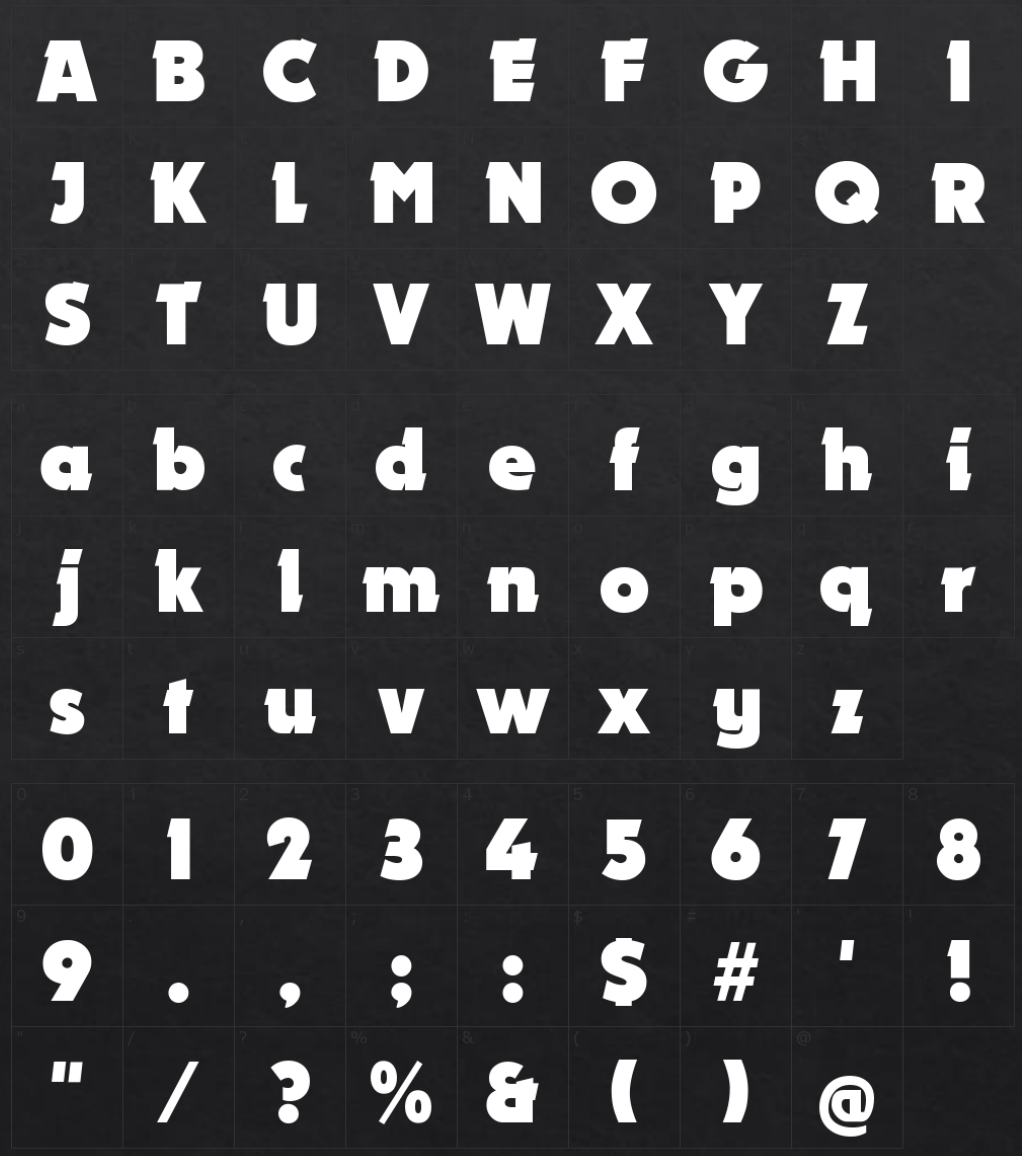
Final Goal

The Tools

- Build System
  - Alex Schwartz Meson build Docker container with gcc13 and nvcc
- Programming
  - Modern C++ based on John Lakos' *Large Scale C++*
  - CUDA 12.6 using best practices from Nvidia
  - PyBind to wrap code into a Python 3.11.2 library
- Third-party Libraries
  - CIMG & SDL2

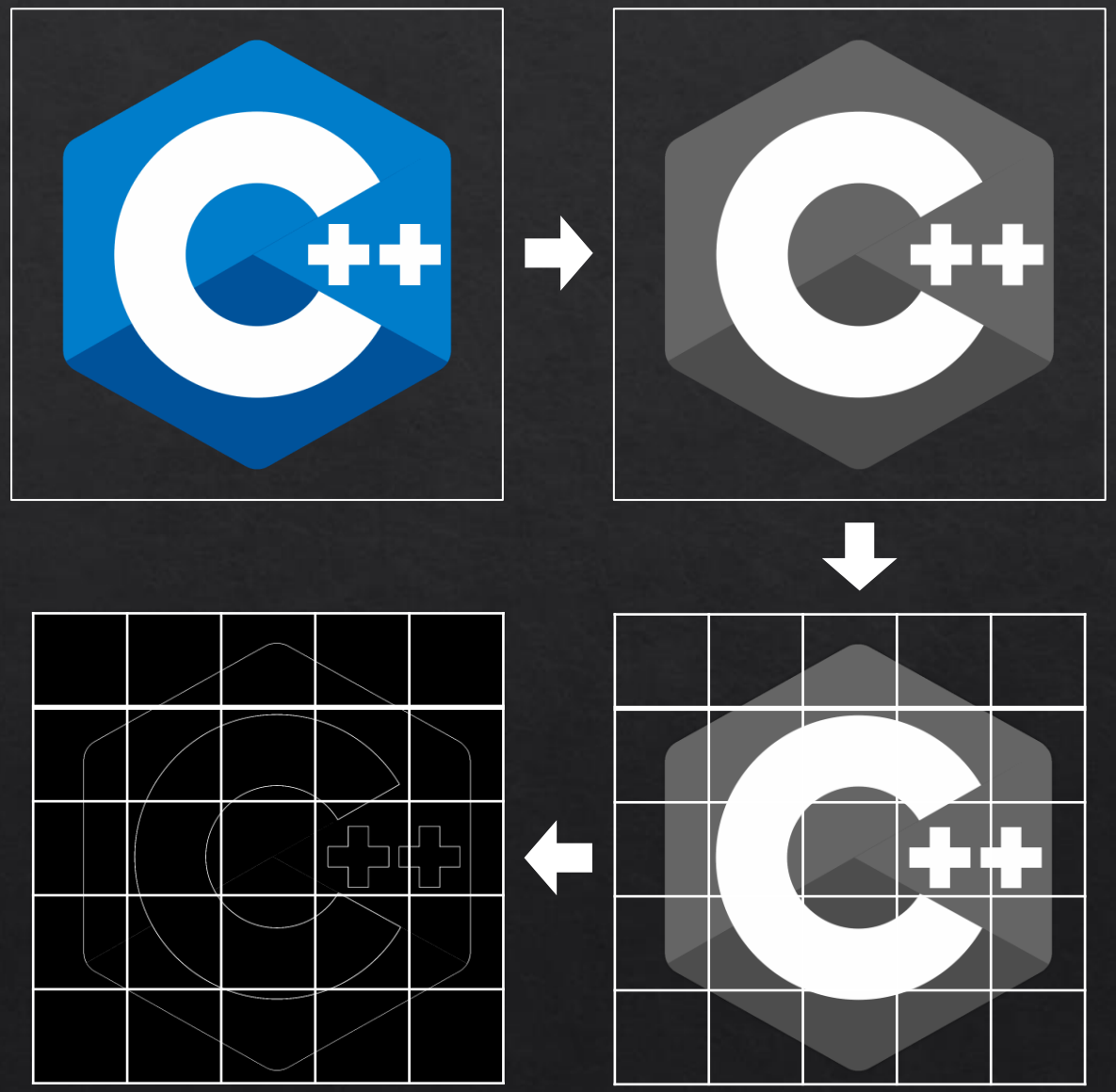
The Preamble

- User provides a path to an image file and font size
- SDL2 draws ASCII characters using the font on new images which define the patch shape



The Preamble

- CIMG does the following
  - Reads the image from the file
  - Converts the image to grayscale
  - Crops the image to fit a whole number of ASCII patches in each dimension
  - Performs a Sobel filter on the image to obtain an edge map
- Edge map is moved to GPU for ASCII conversion



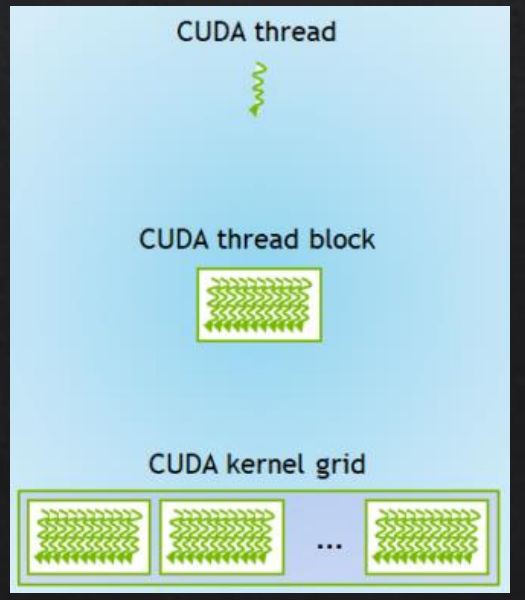
The GPU Code

- An output array is allocated with size that matches the number of patches that in the target image
- CUDA kernel grid dimensions match the dimensions of the output array
- Each CUDA thread block tries to find the ASCII character patch that best matches the patch assigned to the thread block

Output Array

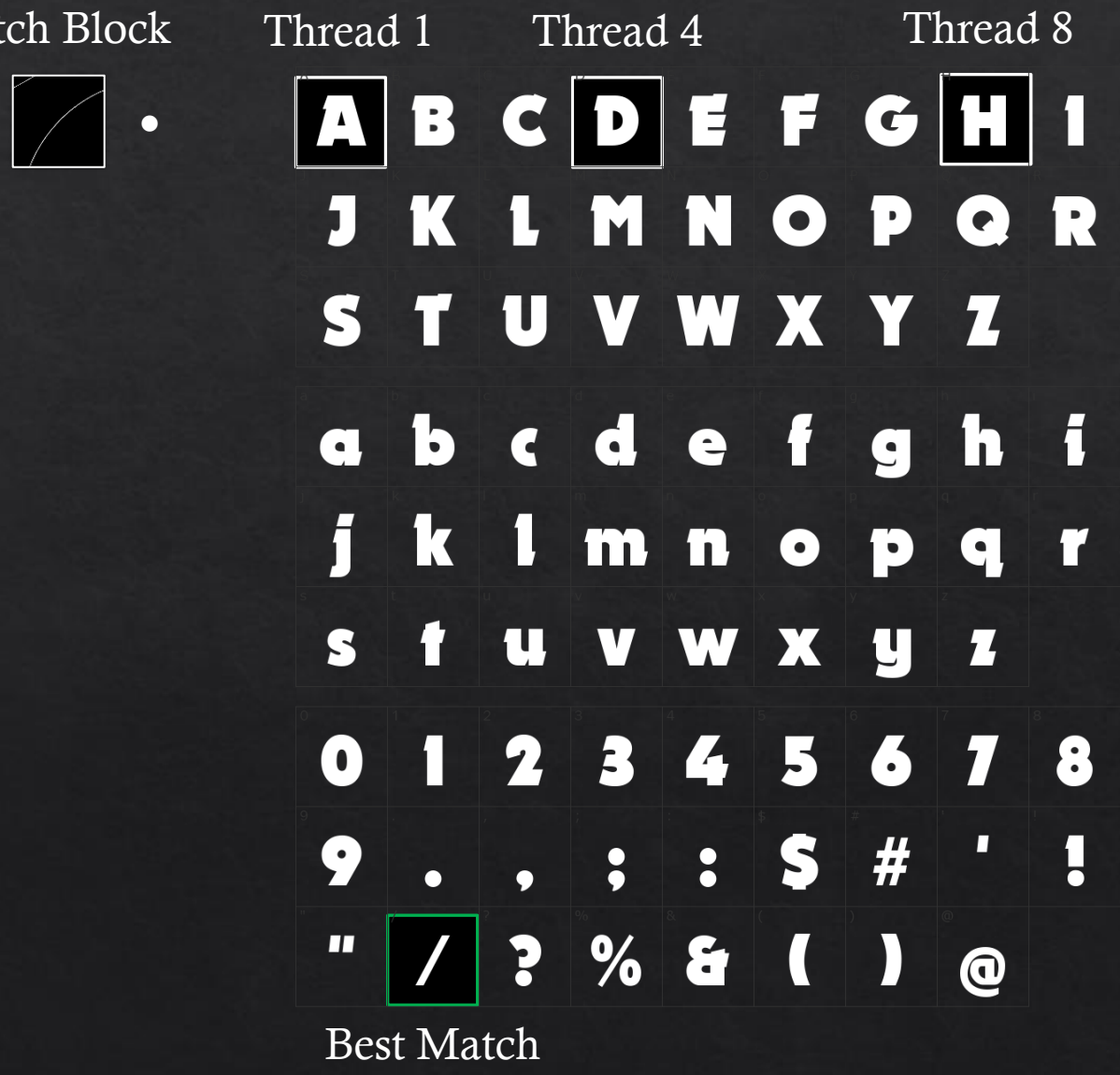
CUDA Kernel Grid

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25



The GPU Code

- Each CUDA thread will perform a **dot product** between the image patch assigned to its block and an ASCII character patch determined by its thread index
- All threads in a block report their similarity metric results so the thread block may assign the **best matching** ASCII character to the output array
- Final output array can be printed to a terminal screen

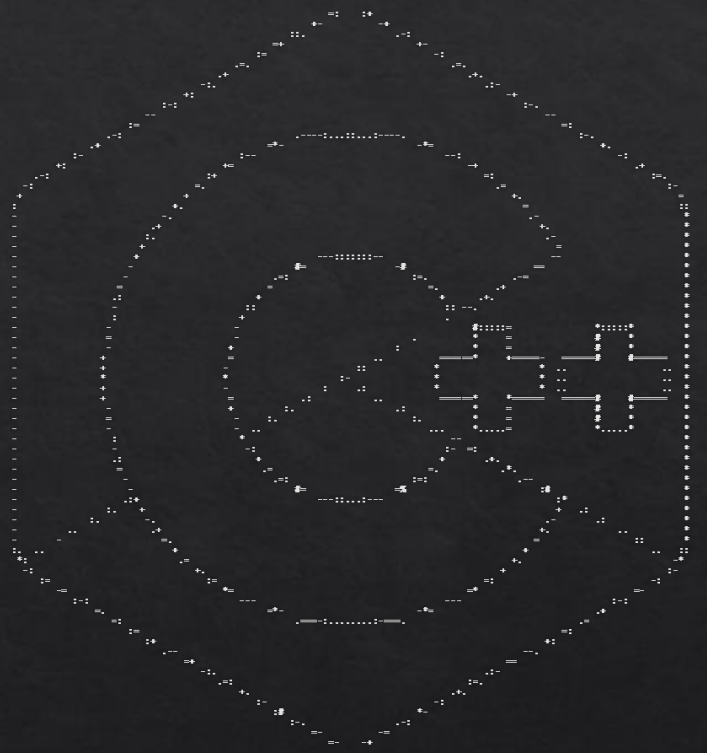


The Result

- All the CUDA kernel block results construct the final ASCII representation
- Using smaller patches via smaller font size, the precision increases

Output Array

		^		
	/	-	\	
|	|		+	|
	\	-	/	
		v		



Future Work

- Allow user to specify image already in memory
- Write a python script to pipe live camera feed into CUDASCII and display it on a terminal with the specified font size

Questions?