DIP Project Presentation

LaTeX Code Generation

from

Printed Equations

Team Escher

Problem

Working with complex, lengthy mathematical equations can be cumbersome

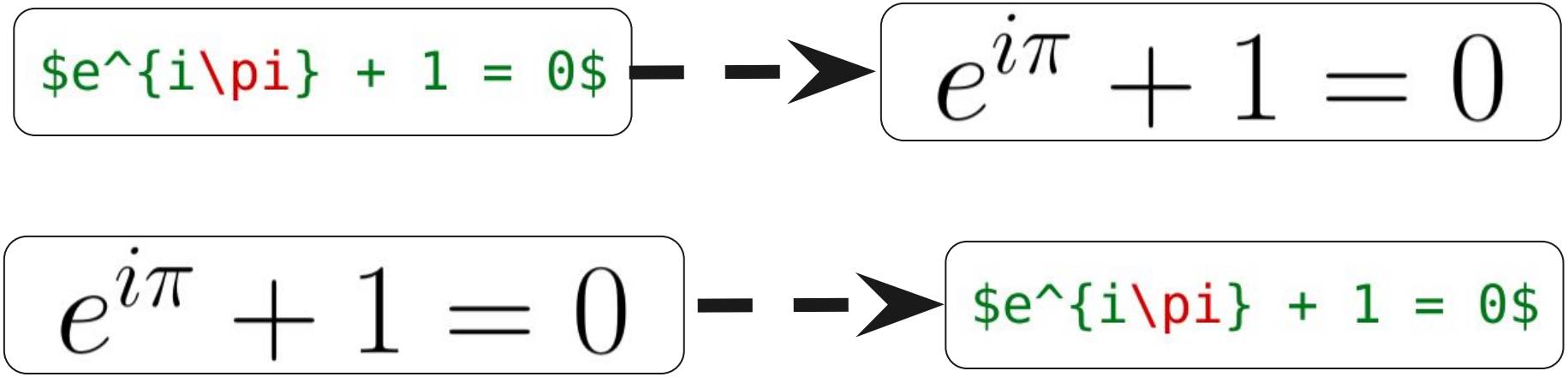
Equations in the printed form are **Not Easily Reproducible** in new LaTeX documents

Overview

1. Introduction
2. Motivation & Scope
3. Solution Pipeline
4. Solution
5. Experiments & Analysis
6. Results
7. Best Solution
8. Conclusion

Introduction

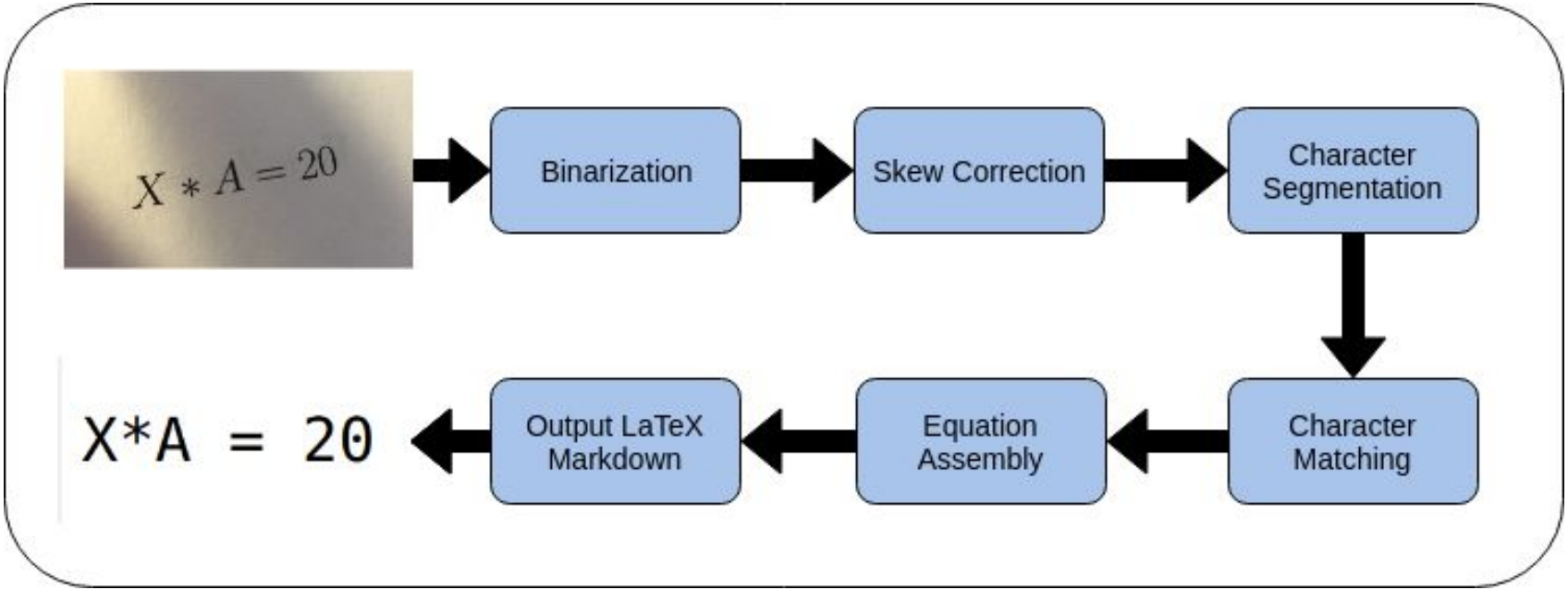
Take a photograph of a mathematical printed equation. We propose a system of converting the image of the equation to LaTeX code.



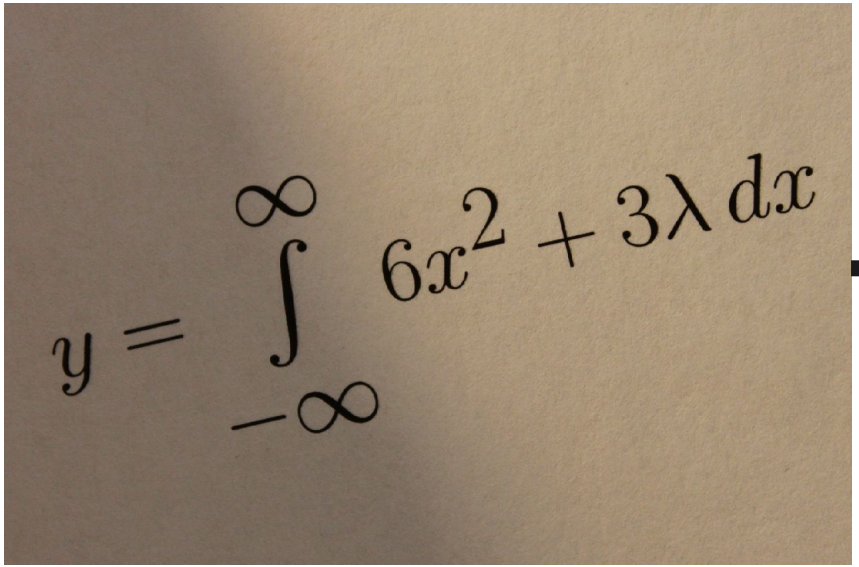
Motivation & *Scope*

- **Motivation**
  LaTeX is a powerful typesetting system that is extremely useful for technical documents
  Large amounts of scientific & mathematical prints are in LaTeX format.
- **Scope**
  Limited by set of mathematical expressions & operations recognized by system
  Input image contains no other text

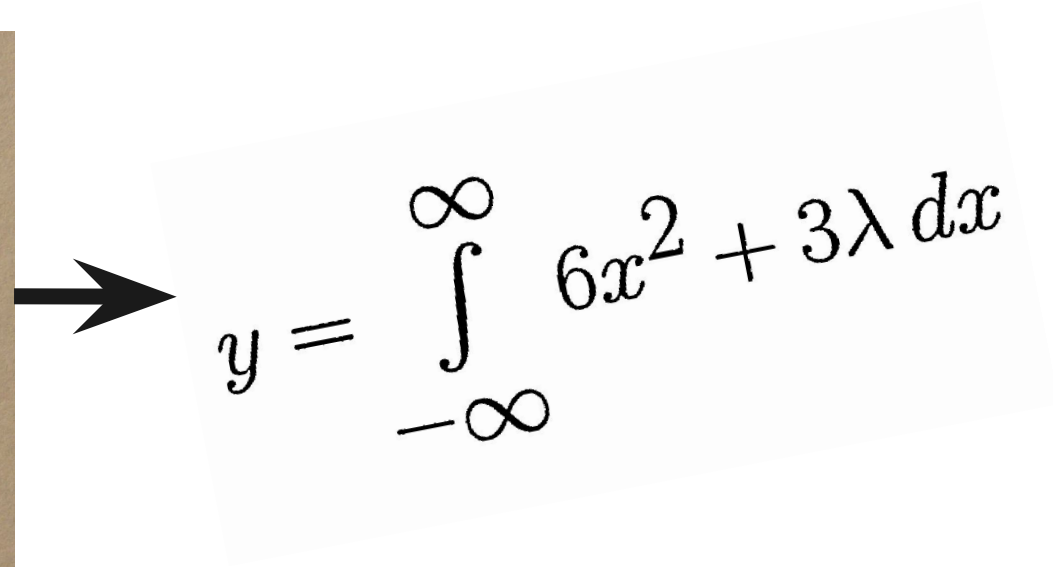
Solution Pipeline



Solution Step 1: *Binarization*



$$y = \int_{-\infty}^{\infty} 6x^2 + 3\lambda \, dx$$



$$y = \int_{-\infty}^{\infty} 6x^2 + 3\lambda \, dx$$

Solution Step 1: *Binarization*

Step 1: Convert Image to grayscale

Step 2: Classify Image - Screenshot Vs Photographed

Case Photograph

Problems: Uneven Lighting, Page Imperfections

Step 3: Resolution Invariance: Remove high frequency noise - Gaussian Blur

Step 4: Heuristic based Adaptive Thresholding (faster than Otsu's method)

Step 5: Inversion: Text becomes foreground

Step 6: Noise Removal (Morphological: Opening + Closing)

Step 7: Small Hole Filling

Solution Step 2: *Skew Correction*

$$y = \int_{-\infty}^{\infty} 6x^2 + 3\lambda \, dx \rightarrow y = \int_{-\infty}^{\infty} 6x^2 + 3\lambda \, dx$$

Solution Step 2: *Skew Correction*

Step 1: Obtain Dominant Orientation Via Hough Transform

Challenges: Long Diagonal Lines (Division Bar)

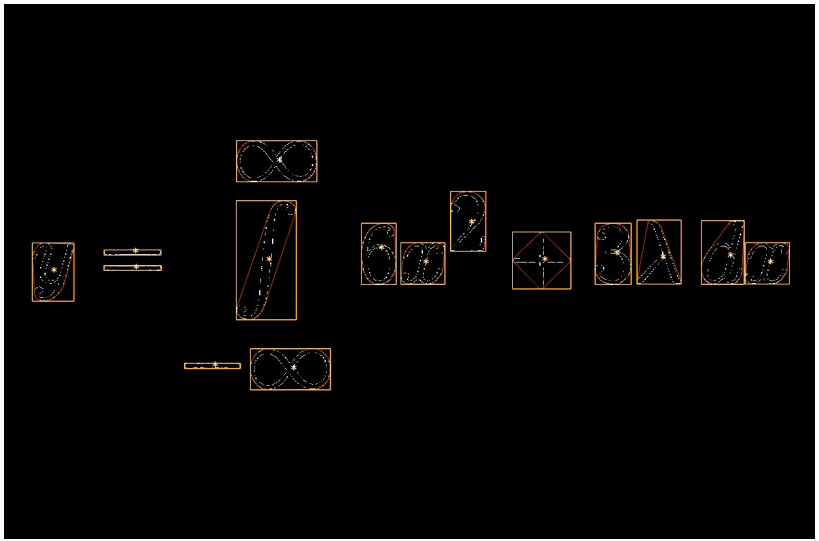
Step2: Take Top 4 Hough Transform peaks

Step 3: Rotate the image

Step 4: Soften edges if angle is large (opening)

Solution Step 3: *Character Segmentation*

$$y = \int_{-\infty}^{\infty} 6x^2 + 3\lambda \, dx$$



Solution Step 3: *Character Segmentation*

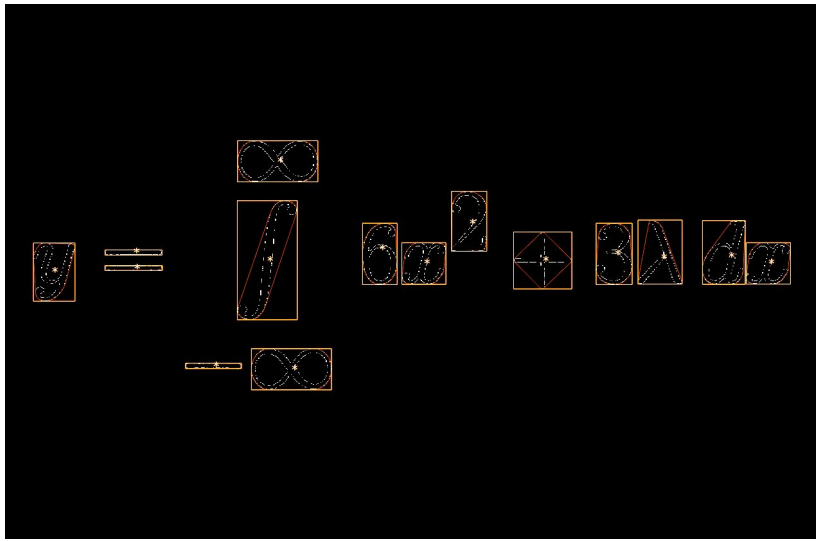
Step 1: Create an **Edge-map** of the input image (erode the inverted image + XOR)

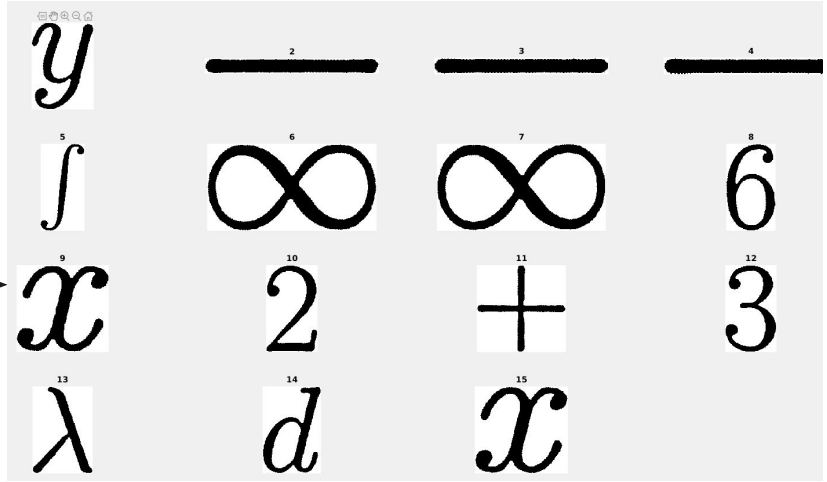
Step 2: Obtain **Centroid, Bounding Box & Convex Hull** of each edge

Useful in the case of square root

Step 3: Select the largest single character in each region

Solution Step 4: *Character Feature Vectors*





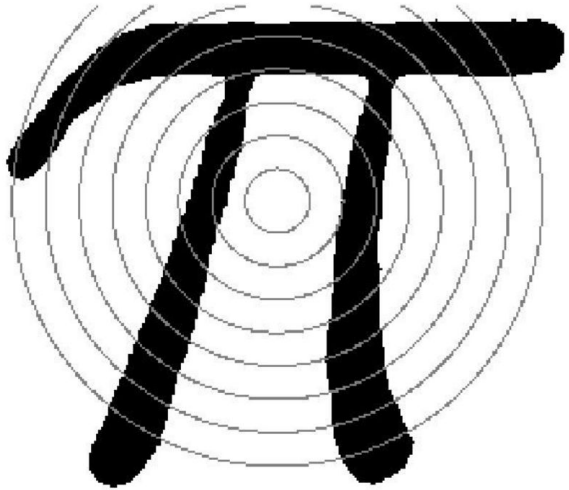
Char < x_1, x_2, … x_{22}>

Solution Step 4: *Character Feature Vectors*

- **22-element** Feature Vector : Identification Profile
- Features are **Invariant to Translation, Scaling & Rotation**

Features 1 - 16:

- **1:** Normalized Central Moment of Inertia
- **2 - 16:** Circular Topology
  - **2 - 9:** Number of times each circle crosses character
  - **10 - 16:** Spacing b/w character crossings for each circle (differentiator)



Solution Step 4: *Character Feature Vectors*

- 22-element character Feature Vector: Identification Profile
- Features are invariant to Translation, Scaling & Rotation

Features 17 - 22:

- Hu Invariant Moments

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \qquad \gamma = 1 + \frac{p+q}{2}$$

$$H_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$H_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$H_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} - \eta_{03})^2$$

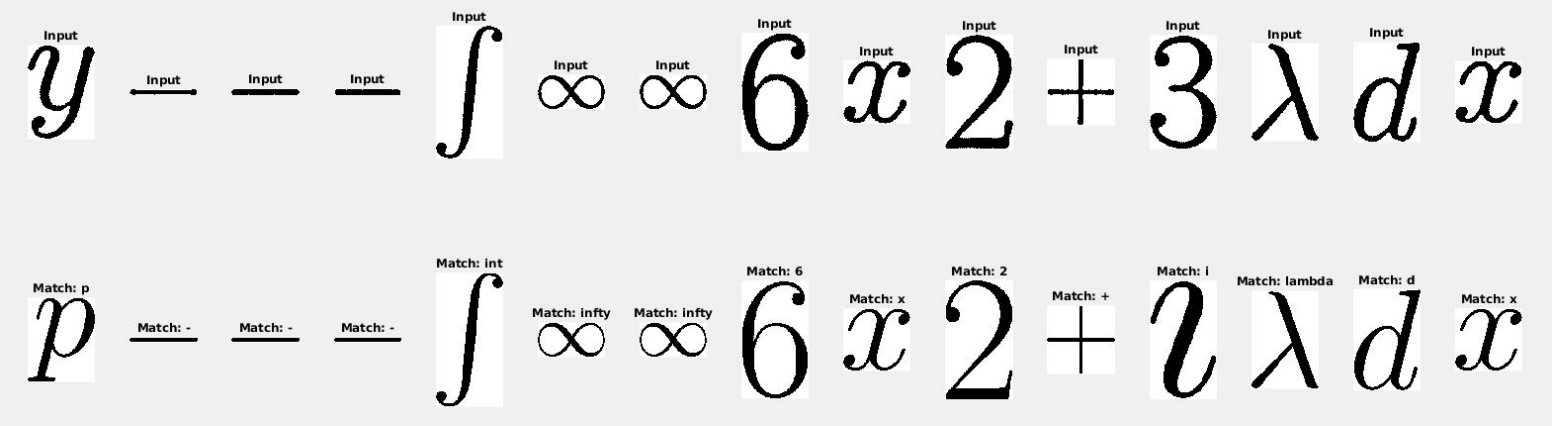
$$H_5 = (\eta_{30} - 3\eta_{12})^2(\eta_{30} + \eta_{12})^2[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

$$H_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{12} + \eta_{03})$$

$$H_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

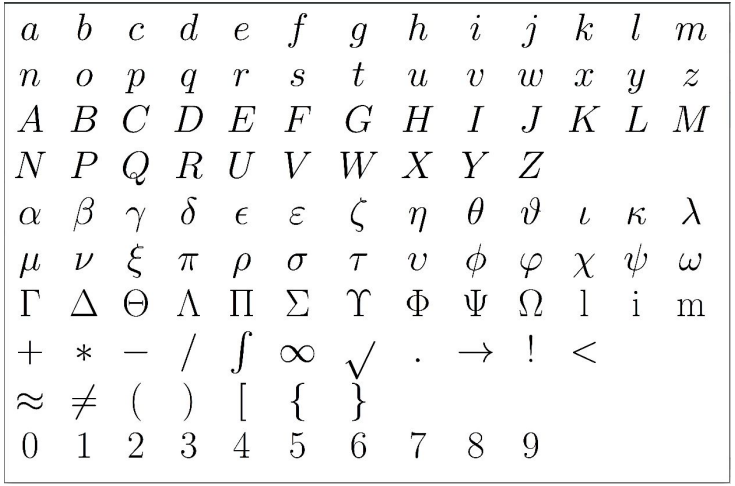
Solution Step 5: *Character Matching*

Char < x_1, x_2, … x_{22}>



Solution Step 5: *Character Matching*

- **Inputs:**
  Character Feature Vector, Database of Target Character Feature Vectors
- **Algorithm:**
  K nearest neighbour Classification
  Distances: Euclidean, Manhattan, Cosine
  Best Performance: **Manhattan**
- **Output:**
  Detected Character Class, Centroid, Bounding Box



Solution Step 6: *Equation Assembly*

Take into account:

- **Left-to-Right** Assembly - Track centroids of characters
- **Subscripts & Superscripts**
- **LaTeX Control Sequences: ‘\theta ‘**
- **Bounding Box Overlaps: ‘=’**
- **Recursive Assembly** (Sub Equations): **Square Root, Fraction bar, Summation**
- (-) Limited number of control sequences supported

Experiments *& Analysis*

- **Heuristic based Adaptive Thresholding** = 2x Speed Otsu's Method
- **Accuracy** of matching directly dependent on **Resolution** of input image
  Scale 1.0 Accuracy : Scale 0.5 Accuracy :: 99 : 87
- **PCA Based Deskewing**: Issue of dominant direction = Diagonal
- **Segmentation** of letter "i": Semantics not considered
- **Low resolution** & poor skew correction are bottlenecks

Experiments *& Analysis*

Image	Correct symbol count	Total symbol count	Percent correct	Reason for Error
Image 1	6	7	85.71	-
Image 2	5	7	71.43	Low resolution
Image 3	5	9	55.55	Poor de-skewing
Image 4	8	11	72.73	Low resolution
Image 5	12	19	63.16	Poor de-skewing
Image 6	13	15	86.67	-
Image 7	10	12	83.33	-
Image 8	19	22	86.36	-
Total	78	102	76.47	-

Table 1: Results for KNN based on Manhattan Distance

Experiments *& Analysis*

Image	Correct symbol count	Total symbol count	Percent correct
Image 1	5	7	71.43
Image 2	5	7	71.43
Image 3	5	9	55.55
Image 4	7	11	63.63
Image 5	12	19	63.16
Image 6	13	15	86.67
Image 7	10	12	83.33
Image 8	14	22	63.63
Total	71	102	69.6

Table 2: Results for KNN based on Euclidean Distance

Experiments *& Analysis*

Image	Correct symbol count	Total symbol count	Percent correct
Image 1	4	7	57.14
Image 2	5	7	71.43
Image 3	4	9	44.44
Image 4	5	11	45.45
Image 5	9	19	47.36
Image 6	11	15	73.33
Image 7	9	12	75
Image 8	6	22	27.27
Total	53	102	51.96

Table 3: Results for KNN based on Cosine Similarity

Experiments *& Analysis*

Image	Equation (LaTeX)	Assembled Equation (LaTeX)
Image 1	y=\omega X+B	y=\omega X+\kappa
Image 2	X*A=20	X*A=i\mu
Image 3	\lambda /25=10\xi	\lambda /25=10\6
Image 4	\frac{x}{2}+\frac{6}{y}=5z	\frac{x}{2}+e=5\Psi
Image 5	6\frac{dx}{dt}+z\frac{dˆ{2}x}{dtˆ{2}}=3x	6\frac{dx}{dt}+z\frac{dˆ{2}x}{dtˆ{2}}=3x
Image 6	y=\int\limits_{-\infty }ˆ{\infty }6xˆ{2}+3\lambda dx	p=\int\limits_{-\infty }ˆ{\infty }6xˆ{2}+3\lambda dx
Image 7	y=\lim\limits_{n\rightarrow 0}\frac{x}{x}	\pi =rightarrowmathrm{i}\frac{x}{x}

Table 4: Results of Equation Assembly

Results

Best Solution for Photographs

- Heuristic Based Adaptive Thresholding
- K Nearest Neighbour Classification
- Manhattan Distance Metric
- Small Skew Angle

Accuracy of Best Solution

Screenshots: **99%**

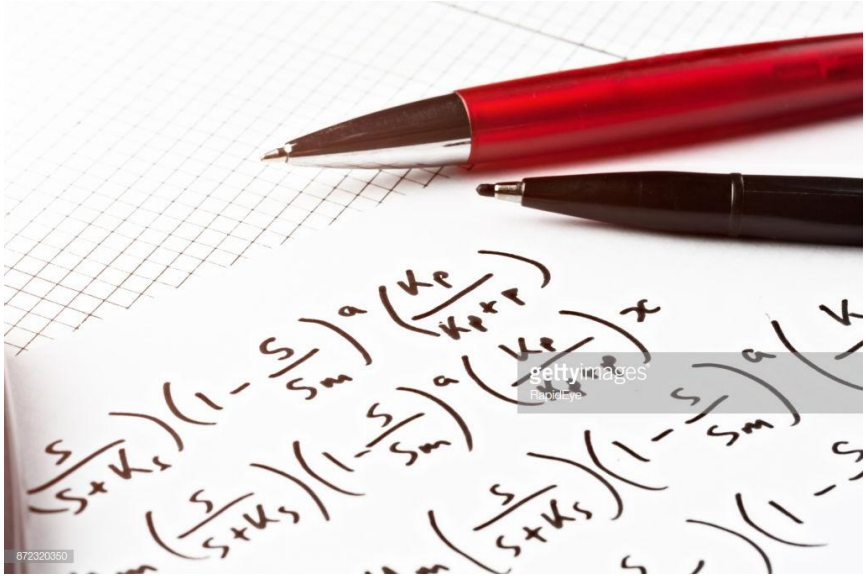
Photograph: **77%**

Input: Photograph of a Printed Equation

Output: Corresponding LaTeX Code

Future *Work*

- Handwritten Equations
- Improve Skew Correction
- Make Pipeline Faster
- Dealing with Low Resolution Images
- Deploy App as a Service



Thank *You*

Team Escher