Perfect Pour: Computer-Vision Powered Visual Feedback for the Optimal Pint of Guinness

Vincent Alesi, Brianna Felipe, Adrian Jackson

ECE 472 Computer Vision & Robotics Final Report

I. Introduction

Guinness is a well-known alcoholic beverage originating from Ireland. One of the most iconic features of this stout beer is its characteristic foam head, which forms at the top of the beer when poured. Since 1959, Guinness has been carbonated with a mixture of nitrogen and carbon dioxide [1]. This mixture produces a smooth texture and thick foam due to the lower solubility and smaller size of nitrogen bubbles when compared to carbon dioxide, the sole carbonating agent in most popular beers.

Correctly pouring a pint of Guinness requires holding the glass and can at the right angle so the amount of foam and liquid reach an ideal ratio. In a bar setting, Guinness is poured from a tap utilizing a double-pour method, which takes over two minutes to serve a single beer. In private settings, people pour from an aluminum can by feel, which can lead to too much or too little foam, known as a flat pour [2]. It's important to note that bottled Guinness relies primarily on carbon dioxide to provide carbonation, a different formulation that produces a significantly flatter and thinner foam [3]. We only considered canned Guinness in our testing, due to ease of procurement and use. The use of nitrogen in tap Guinness results in more unique foam properties, which could prove to be an interesting future experiment.

Our project, Perfect Pour, aims to give real-time feedback on the quality of a Guinness pour. Because Guinness is a dark beer, there is a distinct visual contrast between the foam and liquid, making it an excellent use case for computer vision. We use two medium YOLO models in parallel: a keypoint detection model (for pose estimation) and an instance segmentation model (for glass fill estimation). The system combines both outputs in real time to provide the user with real-time feedback as they pour to optimize their ratio of foam and liquid. Upon the completion of a pour, the system plots the can angle, glass angle and fill history from the pour, showing how the glass and can angles may have contributed to the amount of foam at each stage of the pour. This project is a practical example of applying keypoint detection, instance segmentation, and human-in-the-loop systems to model a novel physical task.

Fig. 1. An image of the ideal Guinness Pour [7].

II. Related Work

Angle detection, clear-object perception, and foam segmentation have been utilized in similar applications, guiding our design choices.

An example bottle-orientation project uses YOLO keypoints to estimate the pose of a bottle. This project is significantly simpler than ours, only predicting the angle as one of a discrete set: 0, 90, or 180 degrees. However, this project validates that angle estimation can be done reliably with a small dataset; in this case, of only 3 images [4]. The bottle angle is determined by calculating the inverse tangent of the difference between the x and

y coordinates of each keypoint. This inspired our approach for determining the tilt of the can and glass.

A significant concern we had prior to undertaking this project was the segmentation of a clear object, i.e., the beer glass. Potential inaccuracies can occur when training a model to detect clear objects, due to their unique translucent properties compared to opaque objects. The ClearPose dataset highlights these difficulties, providing a method to repeatedly detect clear objects despite changing light and background environments. The dataset uses SLAM and 3D reconstruction to create accurate 3D computer models of transparent objects, which provides a given model with foundational knowledge to train upon [5]. Though it is valuable to learn such features which are particular to transparent objects, we determined that this preliminary step was well outside of our project scope. However, we noted the considerations made by the creators of ClearPose. These include the usage of fixed, plain backgrounds and small opaque markers on the glass itself to improve keypoint recognition of the angle model.

Our design was also inspired by prior foam segmentation research. A recent study uses semantic segmentation to separate foam from liquid during the wastewater treatment process. The study achieved a 96% success rate after only 10 epochs utilizing a U-net architecture, even when training images exhibited low contrast between foam and liquid [6]. Though our application is significantly different, using instance segmentation and requiring a much lower latency, this research provided important background for us. In particular, Guinness exhibits high visual contrast between its foam and liquid during pouring, reinforcing the feasibility of this project.

III. Methods

A. System Overview

Our system uses two lightweight YOLOv8 models that work independently in parallel during the pouring process. The first model is a fine-tuned version of YOLOv8m-pose that identifies two key points on the can and two key points on the glass. Using the x and y image coordinates of these points, the system calculates the tilt angle of both objects relative to their position on the captured frame.

The second model is an instance segmentation model, fine tuned from YOLOv8m-seg that separates the inside of the glass into three regions: foam, liquid, and empty space. It locates any instance of one of these three classes, and labels the region of pixels corresponding to them. By extracting the size in pixels of each of these regions, we can effectively measure their size and then calculate the ratio of foam to liquid. Since the desired result of a Guinness pour includes a specific volume of foam at the top of the glass, this ratio is an important part of judging pour quality.

These two models together allow the system to measure pour characteristics in real time and provide relevant feedback to the user. This feedback consists simply of whether to tilt the glass more or less, or maintain the current pour angle. Using such simple feedback is ideal for users because it allows them to quickly react and change their pour.

B. Data Collection

To train the models, we captured multiple videos of an arbitrary Guinness pour with a generic glass of similar shape to official Guinness glasses and a can of Non-Alcoholic Guinness 0. This was done with a python script and Logitech C920 HD Webcam, which enabled high resolution data. The background of this data is plain, with the human pouring typically out of frame to avoid visually interfering with the can add glass. However, because a human must be holding both glass and can throughout the pouring process, their fingers will inevitably obstruct the full view of the object. As such, the data was collected across multiple pours, to organically generate data with natural hand posture that the model would also encounter during its deployment. Additionally, because the data was collected across pours, the glass contained varying levels of liquid and foam. Data was also captured with a variety of angles, lighting conditions, and distances from the camera. Images with and without objects in the same environments were later added, which improved accuracy.

A key concern before data collection was the visual nature of the glass and can. The glass is visually very uniform and completely transparent, making keypoint detection difficult. Because our

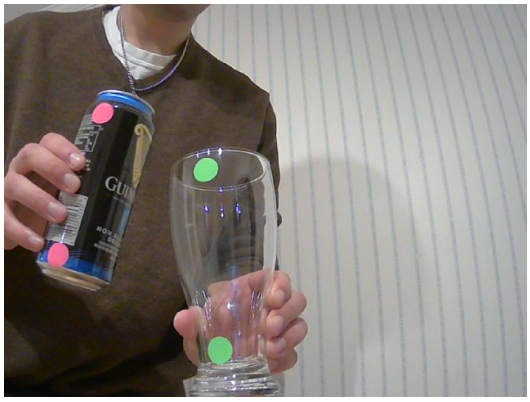

Fig. 2. The can and glass utilized.

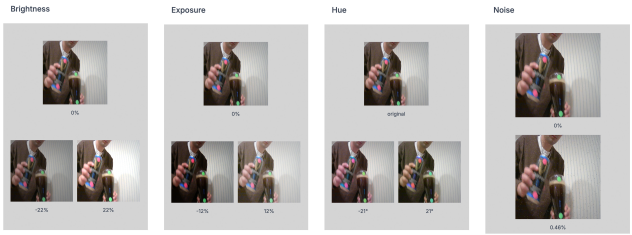

Fig. 3. Image augmentation techniques utilized.

method of angle estimation is quite brittle with regard to the exact placement of each keypoint, we determined that keypoints must be placed exactly on the bisecting line of the face of the glass normal to the camera's perspective. To aid our model in detecting these keypoints and to make the annotating process easier and more consistent, small colorful stickers (Figure 2). Green stickers were used for the glass, and pink for the can. These colors were chosen to be visually distinct from the liquid, can design, and background used in data collection.

This video was sampled at a rate of 5hz using a python script, and any images from this video not depicting the glass or can were discarded. The images were then annotated using Roboflow, a popular platform for computer vision tasks. Two annotations were completed per image: one of the glass and can's pose, and one of the glass' fill state.

The pose annotation was completed by first drawing a bounding box around the can and glass, and labeling it with the appropriate class. Then, a predefined class-specific keypoint skeleton was fine-tuned for that particular image. These skeletons consist of two points; one for the top and another for the bottom of the object. This mirrors the annotations used in the Bottle Orientation blog by Roboflow referenced above. The fill annotation was then completed by drawing pixel masks on the different sections of the glass containing liquid, empty, and foam regions. Annotations were completed by 3 separate annotators, but were standardized by establishing techniques beforehand. Since our dataset is relatively small at 256 total images, we used image augmentation to create higher diversity in training data. This included brightness variation by +/- 15%, exposure variation by +/- 12%, camera gain with a variance of 0.05, and blur of up to 0.4 pixels. Additionally, images are resized to 640x640 for pretrained YOLO model compatibility.

C. Model Training

As stated, two separate machine learning models were trained for this system: a pose estimation model and a fill estimation model. The pose model is a fine-tuned version of YOLOv8m-pose, and the fill model is a fine-tuned version of YOLOv8m-seg. The fill estimation model was trained as an instance segmentation model to classify the contents of the glass as empty, liquid, or foam. The various fill levels present in the dataset enable the model to learn the visual differences between liquid and foam at different fill levels. In particular, the top layer of the liquid portion of a pour tends to be a lighter shade of brown than the bottom due to the continuous settling of carbonation bubbles, which take time to settle. Roboflow was also used as the main platform for training and deployment of both models. Roboflow provides the capability to easily fine-tune publicly available YOLO models with their hardware. This convenience does come at the cost of granularity, however, and we were unable to modify certain training hyperparameters, such as training length and loss function. These two were of particular concern because of their significant impact on model performance. When creating and training a model in Roboflow, the user chooses a model that has a specified size and training length. These cannot be modified (as we are using student accounts), and resulted in gross overfitting of models when allowed to train fully.

There was no automatic early stopping available, so team members monitered training and manually end it when evaluation plateaued.

D. Model Deployment

Both models were deployed independently using Roboflow's hosted inference API and then combined during live operation. For each video frame, the same image is sent to both models, and their outputs are visually combined to generate user feedback. This design allows each model to train on a single task, decreasing model complexity and providing increased deployment flexibility by decoupling our two primary goals. The models are run on inference asynchronously on a live video feed in separate threads within a python script. This script leverages lightweight opencv image capture to increase framerate and decrease latency. A camera is used to capture live video in 640x640 resolution - the same size that YOLO models resize images to. Frames are captured and are sent to two threads for pose and fill calculation. The pose estimator captures the two keypoints of each detected object and computes the arctangent of their y and x differences, therefore computing the object's tilt in degrees. The fill calculator receives a crop of the current frame, cropped to only display the glass, and segments this image into empty, liquid, and foam regions. The proportion of foam to liquid is then calculated and used as an indicator of whether more or less foam is desired in the current pour. If more foam is desired, the user is instructed to increase their tilt; if less, to decrease their tilt. Two specific 'ideal tilts' are predefined as the correct angle to pour at in the cases of a. more foam and b. less foam required. The user is then instructed to increase or decrease their tilt angle to match this ideal angle, depending on their current tilt. The nature of this feedback is basic, and refining it is a goal of future work. For the scope of the project as-is, however, the feedback is sufficient. Confidence thresholds were applied during inference to reduce false detections and improve the stability of the live feedback. This was of particular importance as can detection false positives were common in when testing on non-plain backgrounds. Glass false positives also occurred due to the nature of transparent objects.

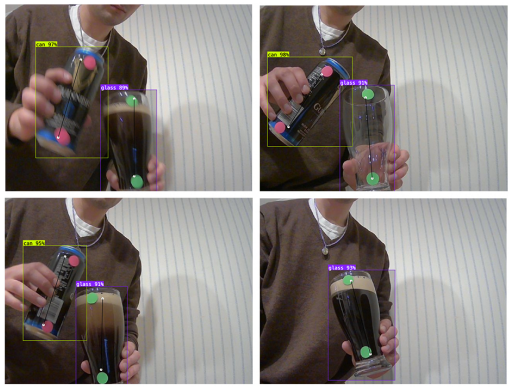

Fig. 4. Detailed training results of the pose model.

E. Evaluation Metric

Because this is a previously unsolved problem, there are scarce metrics to compare our methods to. Generally, we require our model to detect the can and bottle during pours, not detect false positives (a key metric as to avoid incorrect feedback), and detect the correct liquid to foam ratio of the glass. Because these metrics cannot be quantified easily in-situ, save for our trained models, it is difficult to generate ground truth comparisons. We do, however, require the models to operate between 5 and 10 frames per second (preferably on the higher end), so that the user receives timely feedback. 5 frames per second is a base standard in commercial object detection platforms.

IV. Results

A. Training Results

Our group trained multiple models and tested them. The best model was our 11th version of the pose calculator, as earlier versions frequently suffered from incorrect object detection, unstable keypoint placement, and poor real-time performance. The qualitative training results for this final pose model are shown in Figure 4, which demonstrates consistent detection of both the can and the glass, along with stable keypoint localization across different pour angles. Figure 5 shows testing results on still frames with variations in focus, hand posture, object presence, and glass fill level. Together, these figures illustrate that the selected model generalizes well to realistic pouring conditions and maintains reliable pose estimation under common sources of visual variation.

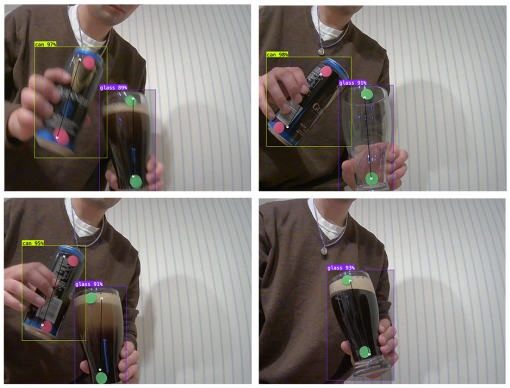

Fig. 5. Results of testing the trained model on still frames with a variety of focus, hand posture, object presence, and glass fill.

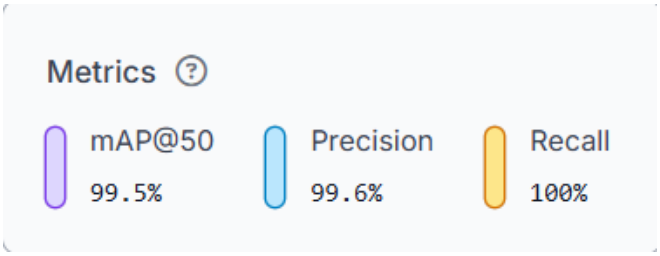

Fig. 6. Metrics shown for the pose model on Roboflow.

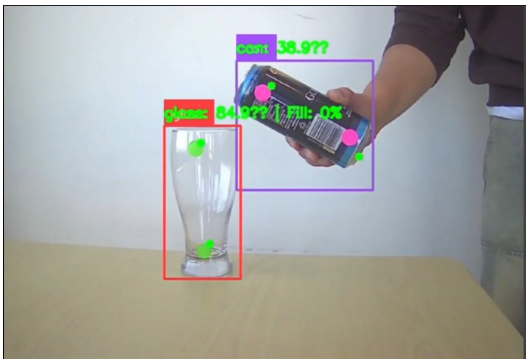

Fig. 7. The controlled environment with a ”dummy” pour.

B. Evaluation Metrics

Although the model reports accuracy values approaching 99-100%, as shown in figure 6, extracting true ground truth performance for keypoint localization is challenging. Roboflow's evaluation metrics primarily check whether the keypoints are in the bounding boxes, rather than measuring pixel-level localization errors. In our dataset, the keypoints are very small, specific, stickers placed on the can and glass. As a result, a specific keypoint may be slightly offset from the true sticker location but still be marked as correct in the evaluation metric. This leads to inflated accuracy values that do not reflect the model's true precision in localizing specific points. Therefore, reported accuracy should be interpreted as strong object-level detection performance rather than keypoint localization accuracy.

The performance of the model and feedback proved difficult to evaluate. There is a lack of precedent for this specific task, and it is difficult to gather ground truth data. The latter would require exact calculation of the angles of the can and glass, either by computer vision (difficult to do and imprecise) or physically while experiments are being conducting (difficult, time consuming, and has potential to disrupt the model's performance.) Thus, we concluded this task was out of scope for this project.

Instead, we opted for more heuristic metrics, focusing on validating whether the model provided high-quality feedback to users. We measured the model's performance in 20 random frames sampled from a video in a controlled, plain environment. We brought the can through a pouring motion without opening it, resulting in a ”dummy” pour. We then counted how many frames the glass, can, and appropriate fill (0%) were correctly or incorrectly identified, how many frames showcased extra objects that were not actually in the scene, and how many frames correctly identified keypoints to a reasonable degree. This was repeated for an actual pour in the same environment. The real pour had significantly lower success rates than the ”dummy” one. Based on these accurate analyses of pours, the system can then generate and provide meaningful feedback. The frame rates of this feedback, however, was significantly lower than 5 frames a second, which does not meet our goal. However, this is likely due to hardware limitations, and better performance can be expected on more powerful hardware.

C. Model Performance

The results of the model in a controlled environment can be seen in table 1, and that of a more realistic environment in table 2.

TABLE I

RESULTS FROM A DUMMY POUR.

Metric	Value
Glass Detection	100%
Can Detection	100%
Fill Correctness	75%
Can Keypoint Correctness	100%
Can Keypoint Correctness	100%

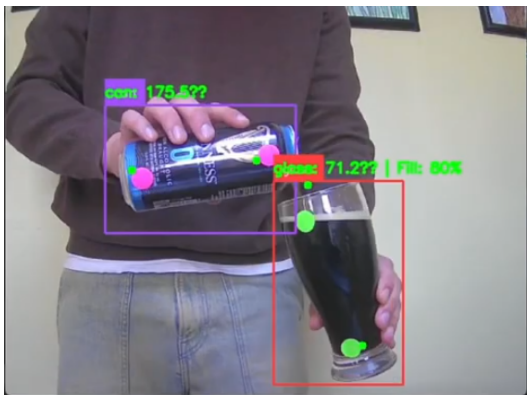

Fig. 8. The controlled environment with a real pour.

D. Ablation Study

For our ablation study, we disabled data augmentation on the dataset prior to training and observed the resultant effect on model performance. This most notably caused the model to detect false positives for the glass, along with other seemingly erratic behavior that likely stemmed from a lack of diverse training data.

Data augmentation assists the model in learning background and viewpoint variation, object-background separation, and contextual consistency. Without it, smooth walls share reflections similar to glass and highlights and gradients trigger detections. This causes the model to lose spatial and contextual understanding. This is especially harmful for keypoint models because keypoints are predicted after detection and false detections create meaningless keypoints.

E. Model Information and Code

The code used for this project has been uploaded to the report submission. A GitHub Repository with the code can be found at https://github.com/adrian-jackson/guinness-calculator.

TABLE II

RESULTS FROM A REAL POUR.

Metric	Value
Glass Detection	100%
Can Detection	75%
Fill Correctness	100%
Can Keypoint Correctness	88%
Can Keypoint Correctness	63%

V. CONCLUSION

In this project, we designed and implemented Perfect Pour, a real-time computer vision system that provides live feedback during the pouring of a Guinness beer. By combining pose estimation and instance segmentation, our system is able to track the angles of the can and glass while also measuring the ratio of foam to liquid throughout the pour. This two-model approach allowed us to decompose a complex physical task into simpler, well-defined vision problems that could be solved efficiently using lightweight YOLO architectures.

Our results demonstrate that YOLO-based models can perform reliably even with relatively small, carefully collected datasets, especially when paired with appropriate data augmentation and constrained environments. The pose estimation model was able to provide stable angle measurements, while the fill estimation model successfully distinguished between foam, liquid, and empty regions in most frames. Although reported accuracy metrics from Roboflow were inflated due to their evaluation method, qualitative testing showed that the system met our real-time performance requirements and provided usable feedback in the majority of pours.

Overall, our project serves as a strong proof of concept for applying real-time computer vision to assist human-performed physical tasks. While the system can be improved through better feedback logic, more diverse data, and testing with real tap-poured Guinness, this project shows that real-time feedback systems are feasible using modern object detection and segmentation models. The techniques explored here could also be extended to other applications involving fluid handling, skill training, or assistive guidance systems.

REFERENCES

[1] Guinness Storehouse, "Frequently Asked Questions," Available: https://www.guinness-storehouse.com/en/faqs. Accessed: December 2025.

[2] Guinness, "Guinness Draught," Available: https://www.guinness.com/en-gb/beers/guinness-draught. Accessed: December 2025.

[3] VinePair, "Everything You Need to Know About Guinness," Available: https://vinepair.com/articles/ntk-guinness/. Accessed: December 2025.

[4] Timothy M, "Estimating Bottle Orientation with Keypoints," Roboflow Blog, 2023.

[5] F. Liu et al., "ClearPose: Large-Scale Transparent Object Dataset and Benchmark," ECCV, 2022.

[6] J. Yang et al., "Semantic Segmentation for Foam Detection in Wastewater Treatment Systems," arXiv:2511.08130, 2025.

[7] Keg King, "How to Serve Guinness on Tap at Home: Your Ultimate Guide," Available: https://www.kegking.com.au/blog/post/how-to-serve-guinness-on-tap-at-home-your-ultimate-guide-with-keg-king. Accessed: December 2025.