Report on Surgical Instrument Segmentation

1. Problem Statement

Accurate segmentation of surgical instruments in real surgical videos is a critical task for computer-assisted interventions. It supports skill assessment, workflow understanding, and potentially decision support systems. However, the problem is challenging due to occlusions, complex backgrounds, and strong class imbalance. The SAR-RARP50 challenge provides a benchmark dataset to explore this problem in a structured way.

Our project aimed to implement a full, reproducible pipeline for SAR-RARP50 under very limited computational resources. The focus was on building a working baseline system rather than pushing for state-of-the-art results.

---

2. Background

The SAR-RARP50 challenge gathered international teams, many of whom used advanced architectures such as SegFormer, Swin Transformer, and Mask2Former, often with ensembles and long training runs on powerful GPUs. Their results showed segmentation scores around 84–85%.

By contrast, our resources were minimal (RTX 3050 GPU 2GB VRAM, temporary cloud credits, limited remote access). With this reality, our goal was to explore the pipeline end-to-end, learn from intermediate steps, and see how far we could reasonably go. The outcome is therefore a constrained but educational baseline.

EDA

This section presents a comprehensive overview of our dataset prior to model development. The goal is to understand its structure, distribution patterns, and quality characteristics, thereby guiding informed design choices for the downstream segmentation model.

Dataset Overview

- **Total Videos: 54**
    - Train: 44 videos (≈ 81.5 %)
    - Test: 10 videos (≈ 18.5 %)
- **Total Frames/Images: 16 295**

- Train: 13 043 (≈ 80 %)
- Test: 3 252 (≈ 20 %)

This split is designed to maintain diversity and ensure that the test set remains unseen and unbiased.

Class Distribution Analysis

- Total Pixels Analyzed: 33.79 billion
- Class-wise Percentages (Overall):

Class	% Pixels	Imbalance Ratio
Background	78.95 %	1.0×
Tool shaft	12.42 %	6.4×
Tool wrist	3.57 %	22.1×
Tool clasper	2.29 %	34.4×
Thread	1.08 %	72.9×
Suturing needle	0.46 %	170.8×
Suction tool	0.63 %	125.5×
Needle Holder	0.18 %	434.5×
Clamps	0.18 %	445.9×
Catheter	0.23 %	343.7×

- **Key Observations**
  - Extreme class imbalance is present, background dominates (≈ 79 %), while several rare instruments (Clamps, Needle Holder, Catheter) are < 0.3 % each.
  - The ratio of background : instruments is ≈ 3.75 : 1.
  - Train vs Test distribution remains largely consistent (differences < 1 % across classes), confirming a stratified-like split.

Image Quality Analysis

- Metrics Computed: Brightness, Contrast, RMS Contrast, Blur Score, Dynamic Range
- Overall Averages:

Metric	Mean	Std	Median	Range (Min–Max)
Brightness	52.15	14.51	51.53	10.40 – 230.37
Contrast	40.61	9.26	40.22	2.24 – 88.12
Blur Score	34.31	14.66	33.09	0.32 – 109.91
Dynamic Range	240.39	5.35	240.00	45 – 255
RMS Contrast	40.61	9.26	40.22	2.24 – 88.12

- **Train vs Test Comparison:**
    - Train images are brighter (+7.9), more contrastive (+4.8), and slightly sharper (+1.7 blur-score) than test images.
    - Dynamic range is stable across splits (≈ 240).

Video-wise Analysis

- **Train Set**
    - Mean frames/video: 296 ± 148 (range 101 – 706)
    - Brightness mean: 53.0 ± 9.4
    - Contrast mean: 41.2 ± 5.0
    - Blur mean: 34.4 ± 9.7
- **Test Set**
    - Mean frames/video: 325 ± 182 (range 117 – 687)
    - Brightness mean: 49.3 ± 11.4
    - Contrast mean: 39.3 ± 7.1
    - Blur mean: 35.9 ± 8.6
- **Per-video heatmaps showed:**

- Certain videos are consistently low in brightness/contrast, while others show high blur (motion artifacts).
- Instrument class composition varies significantly per video, stressing the need for stratified training/validation splitting.

Key Challenges Highlighted by EDA

- Severe Class Imbalance: Minority classes risk underfitting or being ignored by the model.
- Heterogeneous Image Quality: Varying brightness, contrast, and sharpness can introduce domain shifts.
- Background Dominance: High pixel-level presence can bias the model towards predicting background.

Implications for Model Development

- Augment underrepresented classes (oversampling, copy-paste, synthetic compositing).
- Consider class-weighted losses or focal loss.
- Use stratified sampling to maintain class ratios in mini-batches.
- Employ quality-aware filtering or curriculum learning to mitigate low-quality frames during early training.
- Evaluate using per-class metrics (IoU/F1), not only mean scores.

---

3. Methodology

3.1 Pipeline Process

We followed the structured project guide:

- **Step 0:** Project setup with folders/configs.
- **Step 1:** Downloaded dataset archives.
- **Step 2:** Unzipped and reorganized data into per-video folders.
- **Step 3:** Performed class distribution analysis and created RGB masks.
- **Step 4:** Split training/validation with class-aware stratification.

- **Step 5:** Designed augmentation strategy for underrepresented classes (though could not fully use it later due to compute access).
- **Step 6:** Tried multiple models for training.
- **Step 7:** Evaluated validation performance and prepared for test inference.

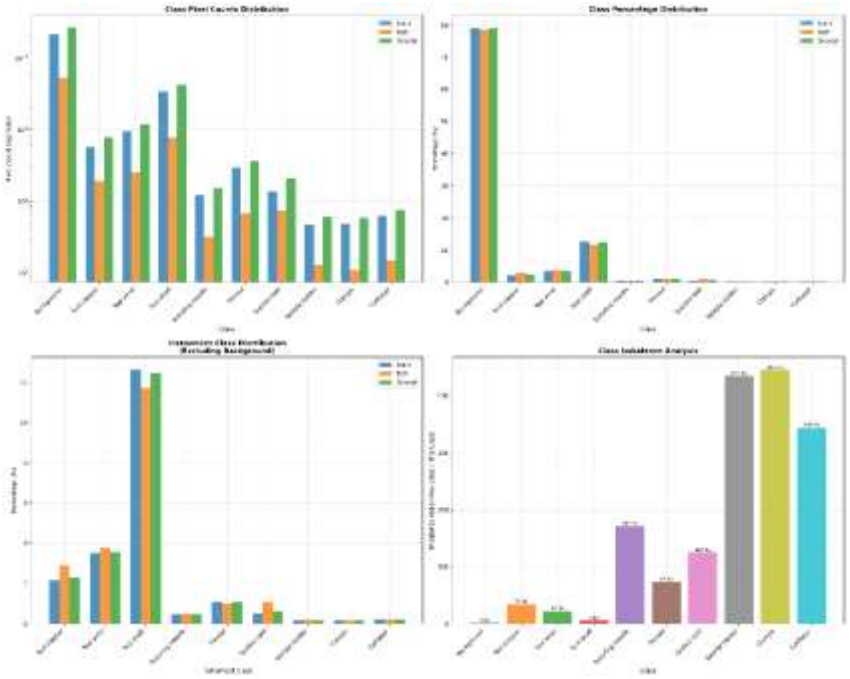
3.2 Model Journey

- **Baseline:** A simple U-Net with attention was tried, but performance was poor and quickly dropped.
- **SAM2 adaptation:** Attempted, but too resource-intensive and unstable.
- **ResNet34 U-Net:** Promising, but training speed (~12 hrs per epoch locally) made it unworkable.
- **Cloud attempts:** RunPod was explored, but interruptions and unstable runtime made it hard to rely on.
- **SegFormer (final attempt):** We settled on **SegFormer**, chosen because it balances transformer-based power with efficiency. On a borrowed system, we managed ~35 epochs (4-5 hours per epoch without class balancing augmentation), but without full augmentations, hyperparameter tuning, or long training schedules.

---

4. Results

4.1 Exploratory Data Analysis (pipeline)



4.2 Training Results

- Approximate validation scores after 35 epochs:
    - **mIoU ~80%**
    - **mNSD ~70%**
- These numbers should be seen only as indicative baselines.

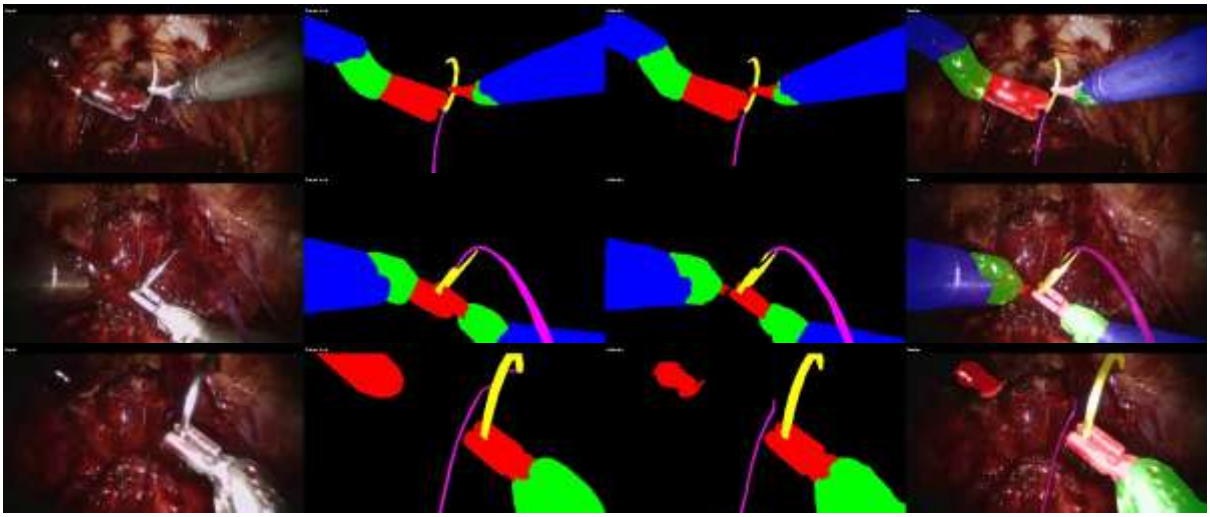
4.3 Test Results

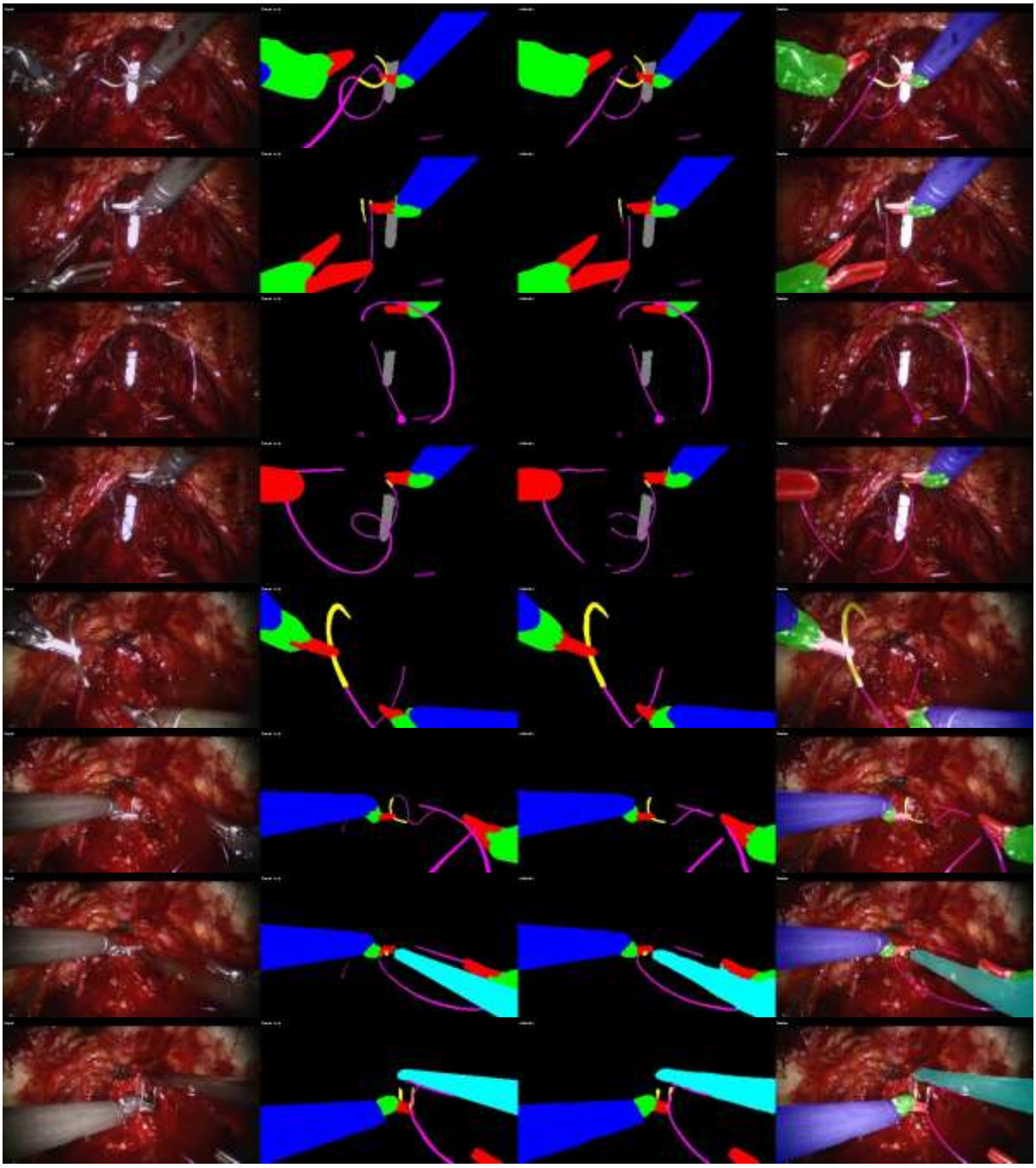
Average mIoU, mNSD, Score =

0.7133 0.697429 0.704701
respectively

video	mIoU	mNSD	score
video_41	0.651247	0.739056	0.693764
video_42	0.774263	0.660484	0.715114
video_43	0.699357	0.722397	0.710784
video_44	0.666332	0.671895	0.669108
video_45	0.69546	0.692818	0.694138
video_46	0.670473	0.694949	0.682601
video_47	0.796564	0.674228	0.732848
video_48	0.755738	0.742413	0.749046
video_49	0.650274	0.627471	0.638771
video_50	0.773289	0.748575	0.760832

4.4 Visualizations

- Example visualizations of test samples
- 



---

5. Interpretation

Our pipeline and experiments confirm that:

- Even without augmentation or tuning, SegFormer learns meaningful patterns.
- However, results are clearly below top-performing challenge submissions (~84–85% segmentation score).

- Our work should be seen as a **baseline attempt under constraints**, not as a final competitive solution.

---

6. Constraints and Limitations

- **Compute:** RTX 3050 could not handle heavier models; 2GB VRAM crashes were frequent.
- **Time:** Training was extremely slow; we had to stop at 35 epochs.
- **Cloud instability:** RunPod interruptions limited sustained progress.
- **No augmentations applied in final run:** Although augmentation design was complete, it was not usable on the remote system (segformer was build remotely)
- **No tuning:** Learning rates, batch sizes, and other hyperparameters were left at defaults.

---

7. Conclusion

We successfully built and tested a complete SAR-RARP50 segmentation pipeline, from raw dataset handling to model training and evaluation. Despite very limited computational resources, we managed to train a SegFormer baseline for 35 epochs, reaching around **80% mIoU** and **70% mNSD** on validation.

This is not a competitive result compared to the top challenge teams (~84–85% segmentation scores), but our contribution lies elsewhere:

- We created a **transparent and reproducible baseline**.
- We identified **clear bottlenecks** in compute, time, and augmentation.
- We laid the foundation for more advanced experimentation once resources allow.

Our work should therefore be understood as a **starting point**, not an endpoint.

---

8. Next Steps and Future Work

Looking ahead, there are two directions we plan to pursue:

1. **Improved Training Setup**
    - Arrange access to stable and higher compute (dedicated cloud GPU or institutional cluster).
    - Fully implement the augmentation pipeline (Step 5).

- Extend training epochs (≥100) and perform systematic hyperparameter tuning.
- Explore more advanced models like Mask2Former or Swin Transformer.

2. **A Unique Frame-Stacking Approach**
   In addition to standard architectures, we are exploring a **temporal-stitching method**:
   - Instead of feeding only frame *t*, the model receives a *stacked 3D-like input* where frame *(t)* is stitched with its immediate predecessor *(t-1)*.
   - The prediction target remains 2D (mask for frame *t*).
   - For the very first frame, we handle it by stitching frame *(0)* with a copy of itself.
   - As an alternative, we also considered computing the **frame difference (t – t-1)** and using it as an attention map to highlight motion and instrument changes.
   - This approach may capture **temporal context** cheaply, without requiring full video transformers. Training would be adapted accordingly, and this direction remains **experimental but promising**.
3. **An eventual learning approach:**
   - Instead of training all the classes, we would focus on underrepresented classes more at first few epochs of training to make model perform similarly better to all classes.

Final Remark

In short, this project demonstrates how much can be achieved with limited resources, while also pointing to the gaps that must be closed to reach competitive performance. The immediate priority is to secure reliable compute power, after which we can systematically extend our pipeline and test our unique temporal-stitching idea. As much as we intend to develop better models using effective yet simple techniques, we couldn't proceed due to computational contraint.
