Research Design Techniques: Data Collection, Modeling & Analysis

1. Research Design Framework

1.1 Overall Research Design

Type: Mixed-Methods Empirical Research with Experimental Validation **Approach**: Quantitative data collection with real-time processing and statistical validation **Paradigm**: Applied Computer Vision and Signal Processing Research

1.2 Research Questions Addressed

1. How accurately can edge-based ML systems detect and classify vehicles in real-time?
2. What is the optimal sensor fusion approach for combining visual and radar data?
3. How do environmental factors affect detection accuracy and system performance?
4. What statistical methods best validate multi-sensor traffic monitoring systems?

2. Data Collection Methodologies

2.1 Primary Data Collection Streams

2.1.1 Visual Data Collection (AI Camera - Sony IMX500)

Data Type: High-resolution video streams (1080p @ 30fps) **Collection Method**: Continuous real-time capture with buffering **Storage Format**:

- Raw frames: RGB arrays (1920×1080×3 pixels)
- Compressed: H.264 encoding for storage efficiency
- Metadata: Timestamp, frame rate, exposure settings, weather conditions

Collection Parameters:

```python
# Camera Configuration
frame_rate = 30  # fps
resolution = (1920, 1080)
buffer_size = 10  # frames
exposure_mode = 'auto'
white_balance = 'auto'
```

Data Quality Assurance:

- Automatic exposure adjustment for varying light conditions
- Frame rate monitoring and dropped frame detection
- Image quality assessment (blur detection, contrast analysis)
- Temporal consistency validation

2.1.2 Radar Data Collection (OPS243-C FMCW Doppler)

Data Type: Doppler frequency measurements **Collection Method**: Serial communication (UART) at 115200 baud **Sampling Rate**: 100 Hz continuous sampling **Data Format**: JSON objects with speed, magnitude, and timestamp

Raw Data Structure:

json

```json
{
  "timestamp": "2025-06-09T14:30:45.123Z",
  "doppler_frequency": 245.7,
  "speed_mph": 35.2,
  "magnitude": 1847,
  "signal_quality": 0.87
}
```

Quality Control Measures:

- Signal-to-noise ratio monitoring
- Outlier detection (speeds >150 mph rejected)
- Temporal smoothing using moving averages
- Magnitude threshold filtering

2.1.3 Environmental Data Collection

Weather Data: API integration with OpenWeatherMap **Collection Frequency**: Every 10 minutes **Parameters Collected**:

- Temperature, humidity, pressure
- Precipitation intensity and type
- Wind speed and direction
- Visibility conditions

- Cloud coverage

System Performance Data:

- CPU utilization and temperature
- Memory usage patterns
- Processing latency measurements
- Network connectivity status
- Power consumption metrics

2.2 Secondary Data Sources

2.2.1 Ground Truth Validation Data

Manual Counting: Human observers for accuracy validation **Reference Speed Measurements**: Police radar gun readings **Traffic Count Devices**: Pneumatic tube counters for baseline comparison **Weather Station Data**: Local meteorological station readings

2.2.2 Historical Context Data

Traffic Patterns: Historical volume and speed data **Incident Reports**: Police and DOT accident/incident databases **Construction Activities**: Road work schedules affecting traffic flow **Event Calendars**: Local events impacting traffic patterns

3. Data Generation and Preprocessing

3.1 Synthetic Data Generation

3.1.1 Augmented Training Data

Purpose: Enhance ML model robustness **Techniques Used**:

- Geometric transformations (rotation, scaling, translation)
- Photometric adjustments (brightness, contrast, saturation)
- Weather simulation (rain, fog, snow effects)
- Lighting condition variations (dawn, dusk, nighttime)

Implementation:

python

```python
# Data Augmentation Pipeline
augmentation_pipeline = [
    RandomBrightness(limit=0.3),
    RandomContrast(limit=0.3),
    RandomRotation(limit=5),
    RandomScale(scale_limit=0.1),
    WeatherAugmentation(['rain', 'fog', 'snow'])
]
```

3.1.2 Simulation Data

Traffic Scenario Simulation: Varied vehicle types, speeds, and densities **Weather Condition Modeling**: Statistical weather pattern generation **Edge Case Generation**: Unusual scenarios for robustness testing

3.2 Data Preprocessing Pipeline

3.2.1 Image Preprocessing

Noise Reduction: Gaussian filtering and bilateral filtering **Normalization**: Pixel value scaling (0-255 → 0-1) **Region of Interest**: Dynamic ROI selection based on traffic lanes **Motion Detection**: Background subtraction for vehicle identification

Processing Steps:

1. Frame capture and buffer management
2. Noise reduction and enhancement
3. ROI extraction and normalization
4. Feature extraction preparation

3.2.2 Radar Data Preprocessing

Signal Filtering: Low-pass filtering for noise reduction **Outlier Removal**: Statistical outlier detection (Z-score > 3) **Temporal Smoothing**: Moving average with configurable window **Calibration**: Speed measurement calibration against known references

Filtering Algorithm:

python

```python
def filter_radar_data(raw_data, window_size=5):
    # Remove statistical outliers
    z_scores = np.abs(stats.zscore(raw_data))
    filtered_data = raw_data[z_scores < 3]

    # Apply moving average
    smoothed_data = np.convolve(filtered_data,
                               np.ones(window_size)/window_size,
                               mode='valid')
    return smoothed_data
```

4. Modeling Approaches and Techniques

4.1 Computer Vision Models

4.1.1 Vehicle Detection Models

Primary Model: YOLOv8 (You Only Look Once)

- **Architecture**: Convolutional Neural Network with anchor-free detection
- **Input**: 640×640 RGB images
- **Output**: Bounding boxes with confidence scores and class labels
- **Classes**: Car, truck, motorcycle, bus, emergency vehicle

Model Configuration:

yaml

```yaml
model:
  backbone: CSPDarknet53
  neck: PANet
  head: YOLOv8Head
  num_classes: 5
  input_size: [640, 640, 3]
  confidence_threshold: 0.5
  nms_threshold: 0.4
```

Alternative Models (for comparison):

- **MobileNetV3**: Lightweight model for resource-constrained environments
- **EfficientDet**: Balanced accuracy-efficiency trade-off

- **Background Subtraction**: Traditional computer vision fallback

4.1.2 Vehicle Tracking Models

SORT Algorithm (Simple Online and Realtime Tracking):

- **Kalman Filter**: Motion prediction and state estimation
- **Hungarian Algorithm**: Optimal assignment of detections to tracks
- **Track Management**: Track initialization, maintenance, and deletion

Tracking Pipeline:

1. Detection association with existing tracks
2. Kalman filter prediction and update
3. Track state management (confirmed, tentative, deleted)
4. Unique ID assignment and persistence

4.2 Sensor Fusion Models

4.2.1 Data Association Model

Probabilistic Approach: Bayesian inference for vehicle-speed matching **Distance Metrics**:

- Spatial proximity (vehicle position vs. radar beam angle)
- Temporal correlation (detection time vs. radar measurement time)
- Velocity consistency (visual motion vs. radar speed)

Association Algorithm:

python

```python
def associate_detections(visual_tracks, radar_measurements):
    association_matrix = calculate_association_costs(
        visual_tracks, radar_measurements)

    # Hungarian algorithm for optimal assignment
    row_indices, col_indices = linear_sum_assignment(association_matrix)

    return create_associations(row_indices, col_indices)
```

4.2.2 Kalman Filter Implementation

State Vector: [x, y, vx, vy, width, height, speed] **Process Model**: Constant velocity motion model **Measurement Model**: Combines visual position and radar speed

Filter Equations:

- **Prediction**: $\hat{x}(k|k-1) = F \cdot \hat{x}(k-1|k-1)$
- **Update**: $\hat{x}(k|k) = \hat{x}(k|k-1) + K \cdot (z(k) - H \cdot \hat{x}(k|k-1))$
- **Kalman Gain**: $K = P(k|k-1) \cdot H^T \cdot (H \cdot P(k|k-1) \cdot H^T + R)^{(-1)}$

4.3 Anomaly Detection Models

4.3.1 Statistical Process Control

Control Charts: Shewhart charts for traffic flow monitoring **Statistical Limits**: 3-sigma control limits for anomaly detection **Trend Analysis**: Moving averages and exponential smoothing

4.3.2 Machine Learning Anomaly Detection

Isolation Forest: Unsupervised outlier detection **One-Class SVM**: Novelty detection for unusual traffic patterns **Autoencoders**: Deep learning approach for pattern recognition

5. Statistical Methods and Analysis

5.1 Descriptive Statistics

5.1.1 Traffic Flow Metrics

Volume Statistics:

- Vehicles per hour (mean, median, standard deviation)
- Peak hour factors and directional distributions
- Vehicle classification percentages

Speed Statistics:

- Speed distributions (histograms, percentiles)
- 85th percentile speeds for traffic engineering
- Speed variance and coefficient of variation

Temporal Analysis:

- Hourly, daily, and seasonal traffic patterns
- Peak period identification and characterization

- Holiday and special event impact analysis

5.2 Inferential Statistics

5.2.1 Hypothesis Testing

Detection Accuracy Testing:

- **Null Hypothesis**: Detection accuracy ≤ 90%
- **Alternative Hypothesis**: Detection accuracy > 90%
- **Test Statistic**: Proportion test (z-test)
- **Significance Level**: $\alpha = 0.05$

Speed Measurement Validation:

- **Paired t-test**: Radar measurements vs. ground truth
- **Confidence Intervals**: 95% CI for measurement accuracy
- **Effect Size**: Cohen's d for practical significance

5.2.2 Correlation Analysis

Sensor Correlation:

- Pearson correlation between visual and radar measurements
- Spearman rank correlation for non-parametric analysis
- Cross-correlation for temporal alignment

Environmental Impact Analysis:

- Correlation between weather conditions and detection accuracy
- Multiple regression for multi-factor analysis
- ANOVA for categorical weather condition effects

5.3 Advanced Statistical Methods

5.3.1 Time Series Analysis

Traffic Flow Modeling:

- ARIMA models for traffic volume prediction
- Seasonal decomposition (STL decomposition)
- Fourier analysis for periodic pattern identification

Change Point Detection:

- CUSUM (Cumulative Sum) control charts
- Bayesian change point detection
- Structural break testing

5.3.2 Multivariate Analysis

Principal Component Analysis (PCA):

- Dimensionality reduction for multi-sensor data
- Feature importance identification
- Data visualization and clustering

Factor Analysis:

- Latent factor identification in traffic patterns
- Common factor extraction
- Factor rotation for interpretability

5.4 Performance Evaluation Metrics

5.4.1 Classification Metrics

Detection Performance:

- **Precision**: TP/(TP+FP) - accuracy of positive predictions
- **Recall**: TP/(TP+FN) - completeness of positive predictions
- **F1-Score**: 2×(Precision×Recall)/(Precision+Recall)
- **mAP**: Mean Average Precision across all classes

Confusion Matrix Analysis:

- True/False Positive/Negative rates
- Class-specific performance analysis
- Error pattern identification

5.4.2 Regression Metrics

Speed Measurement Accuracy:

- **Mean Absolute Error (MAE)**: Average absolute deviation

- **Root Mean Square Error (RMSE)**: Penalizes larger errors
- **Mean Absolute Percentage Error (MAPE)**: Relative error measure
- **R-squared**: Coefficient of determination

5.4.3 System Performance Metrics

Real-time Performance:

- Processing latency distributions
- Throughput measurements (frames/second)
- Resource utilization statistics
- System uptime and reliability metrics

6. Data Validation and Quality Assurance

6.1 Cross-Validation Techniques

6.1.1 Temporal Cross-Validation

Time-Based Splits: Train on historical data, test on future data **Rolling Window Validation**: Sliding window approach for time series **Seasonal Validation**: Training on one season, testing on another

6.1.2 Spatial Cross-Validation

Location-Based Splits: Different traffic monitoring locations **Environmental Condition Splits**: Various weather and lighting conditions **Traffic Density Splits**: Low, medium, and high traffic scenarios

6.2 Statistical Validation Methods

6.2.1 Bootstrap Methods

Confidence Interval Estimation: Bootstrap sampling for metric uncertainty **Hypothesis Testing**: Permutation tests for statistical significance **Model Stability**: Bootstrap aggregation for robust estimates

6.2.2 Monte Carlo Methods

Simulation Studies: Monte Carlo simulation for system performance **Sensitivity Analysis**: Parameter perturbation studies **Uncertainty Quantification**: Probabilistic performance estimates

7. Data Analysis Pipeline

7.1 Real-time Analysis

Stream Processing: Continuous data analysis as data arrives **Online Statistics**: Incremental mean, variance, and quantile updates **Adaptive Thresholds**: Dynamic threshold adjustment based on current conditions

7.2 Batch Analysis

Daily Reports: Comprehensive traffic pattern analysis **Weekly Trends**: Comparative analysis across time periods **Monthly Summaries**: Long-term trend identification and reporting

7.3 Anomaly Analysis

Real-time Alerts: Immediate notification of unusual patterns **Root Cause Analysis**: Statistical investigation of anomalies **Trend Analysis**: Long-term pattern changes and their implications

This comprehensive research design framework ensures robust data collection, sophisticated modeling approaches, and rigorous statistical validation for the traffic monitoring system, providing both real-time operational capabilities and research-quality analytical insights.