ЛЕКЦИЯ № 13 · Проект MATLAB IoT

# Raspberry Pi, веб-камеры и ThingSpeak для анализа трафика на оживлённой трассе

**Бельгибаев Бауржан Абдрахимович**
Д.т.н., доцент
Кафедра «Искусственный интеллект и big data»
ОП 6B07113 «Интеллектуальные системы управления»

# Цель лекции

Изучить архитектуру системы видеомониторинга дорожного потока на базе Raspberry Pi, веб-камеры, MATLAB и ThingSpeak; освоить основные методы выделения движущихся объектов, подсчёта транспортных средств и передачи агрегированных показателей в облако с учётом ограничений точности, производительности и приватности.

# План лекции

1. Постановка задачи анализа дорожного потока
2. Архитектура аппаратно-программной системы
3. Выбор камеры и геометрия установки
4. Получение видеокадров и предварительная обработка
5. Выделение движущихся объектов
6. Морфологическая обработка и связные компоненты
7. Отслеживание объектов и пересечение контрольной линии
8. Интенсивность, плотность, скорость и занятость
9. Реализация прототипа в MATLAB
10. Публикация агрегированных показателей в ThingSpeak
11. Погрешности, валидация и вычислительные ограничения
12. Безопасность, приватность и план эксперимента

# 1 Постановка задачи анализа дорожного потока

Интеллектуальные транспортные системы используют данные о движении автомобилей для оценки загруженности дорог, выявления заторов и планирования транспортной инфраструктуры. Видеокамера позволяет наблюдать несколько полос одновременно, однако преобразование изображения в количественные показатели требует отдельного алгоритма компьютерного зрения.

В рамках учебного проекта требуется определить количество автомобилей, пересекших контрольную линию за интервал наблюдения, вычислить интенсивность потока и отобразить её изменение во времени. Дополнительными показателями могут быть занятость участка, средняя скорость и распределение объектов по классам, если алгоритм обеспечивает их достоверное определение.

Принципиально важно различать обнаружение автомобиля на кадре и подсчёт уникальных автомобилей. Если один объект виден в 30 кадрах, простая сумма обнаружений ошибочно даст 30 машин. Для корректного подсчёта необходимо отслеживать объекты между кадрами или фиксировать события пересечения линии.

# 1 Постановка задачи анализа дорожного потока (продолжение)

Термин «оживлённая трасса» предполагает сложные условия: перекрытие автомобилей, разные скорости, изменение освещения, тени, осадки и вибрацию камеры. Поэтому демонстрационный алгоритм на синтетических кадрах не следует считать готовым промышленным решением.

В данной лекции акцент сделан на методологии и базовой реализации. Более сложные вопросы устойчивого трекинга, оценки качества и расширения системы рассматриваются в продолжении темы — лекции №14.

2

# Архитектура системы Raspberry Pi — камера — MATLAB — ThingSpeak

Структурная цепочка включает веб-камеру, Raspberry Pi, локальную программу обработки видеопотока, модуль расчёта показателей, сетевой интерфейс и канал ThingSpeak. MATLAB применяется для разработки, анализа алгоритмов и визуализации результатов; выполнение на самом Raspberry Pi зависит от выбранного программного окружения и совместимости инструментов.

На краевом устройстве предпочтительно выполнять обработку кадров и публиковать только агрегированные показатели: число автомобилей за минуту, среднюю оценку скорости и статус качества измерений. Такой подход уменьшает объём передаваемых данных и риски раскрытия информации о водителях.

Облачный сервис ThingSpeak удобен для хранения временных рядов, но не предназначен для передачи полного видеопотока как последовательности кадров. В канал следует передавать числовые признаки, а исходные видео хранить локально только при наличии правовых оснований и необходимости.

Для корректной архитектуры задают период захвата кадров, частоту вычисления метрик, период публикации, правила восстановления после потери сети и предельную загрузку процессора. Период публикации может быть значительно больше периода обработки кадров.

# 2 Архитектура системы Raspberry Pi — камера — MATLAB — ThingSpeak (продолжение)

Критичные к времени решения не должны зависеть от доступности облачного сервиса. ThingSpeak в этой схеме выполняет функции мониторинга и аналитики, а не непосредственного управления дорожными светофорами.

# 3 Камера, место установки и геометрия наблюдения

На качество распознавания влияют разрешение камеры, частота кадров, выдержка, поле зрения и угол установки. Слишком низкое разрешение затрудняет выделение удалённых автомобилей; слишком низкая частота кадров увеличивает вероятность пропуска быстрого объекта.

Камеру следует размещать неподвижно с обзором выбранной контрольной линии. При перспективном изображении размер автомобиля зависит от расстояния до камеры, поэтому единый порог площади объекта может работать неодинаково в разных областях кадра.

Для оценки реальной скорости требуется калибровка геометрии: известное расстояние между контрольными линиями, преобразование координат изображения в плоскость дороги либо иная метрическая привязка. Без калибровки можно оценивать скорость в пикселях в секунду, но нельзя обоснованно переводить её в км/ч.

При видеосъёмке в публичном пространстве необходимо учитывать местное законодательство, правила доступа к дорожной инфраструктуре и защиту персональных данных. Для учебной отладки предпочтительны собственные постановочные видео без распознаваемых лиц и номеров либо разрешённые открытые наборы данных.

# 3 Камера, место установки и геометрия наблюдения (продолжение)

Перед началом эксперимента фиксируют положение камеры, размеры кадра, частоту кадров, время суток, погоду, число полос и направление движения. Без этих метаданных результаты трудно воспроизвести.

# 4 Получение кадров и предварительная обработка

Цифровое видео представляется последовательностью кадров I(x,y,t). Для цветного изображения каждый пиксель содержит три компоненты RGB; для базового обнаружения движения часто используется изображение яркости Y(x,y,t). Перевод в оттенки серого уменьшает вычислительную нагрузку.

Предварительная обработка включает изменение размера кадра, выделение области интереса ROI, подавление шума и коррекцию освещения. Уменьшение разрешения ускоряет расчёты, но может приводить к потере небольших объектов. Выбор параметров — компромисс между точностью и производительностью.

ROI ограничивает обработку дорожным полотном и исключает несущественные области, например небо, тротуары и здания. Маска ROI должна соответствовать фактической геометрии сцены, иначе автомобиль может частично исчезнуть из анализируемой области.

При резком изменении освещения простой алгоритм разности кадров способен ошибочно классифицировать большую часть изображения как движение. Для устойчивости используют адаптивную модель фона, фильтрацию и проверку формы обнаруженных областей.

На Raspberry Pi важно измерять реальную скорость обработки кадров и нагрузку на процессор. Значение FPS видеокамеры не равно FPS алгоритма: программа может успевать анализировать только часть поступающих кадров.

# 5 Выделение движущихся объектов

**Один из простейших методов —** разность между текущим кадром $I_t$ и фоновым изображением $B_t$. Вычисляется $D_t(x,y)=|I_t(x,y)-B_t(x,y)|$, после чего бинарная маска движения определяется правилом $M_t(x,y)=1$ при $D_t(x,y)>\tau$ и $M_t(x,y)=0$ в остальных случаях.

Если фон постоянный, движущиеся автомобили выделяются достаточно просто. Однако при изменении освещения, тенях и колебаниях камеры появляется много ложных пикселей. Порог $\tau$ подбирается экспериментально по размеченным кадрам.

Адаптивная модель фона может обновляться по формуле $B_t=(1-\alpha)B_{t-1}+\alpha I_t$, где $0<\alpha<1$ — коэффициент обновления. Слишком большое $\alpha$ приводит к быстрому включению медленных объектов в фон, а слишком малое — к медленной адаптации к освещению.

Разность соседних кадров выделяет изменившиеся области, но часто создаёт двойные контуры вокруг объекта. Для подсчёта машин предпочтительно использовать фоновые модели или обученные детекторы, если доступны достаточные вычислительные ресурсы.

# 5 Выделение движущихся объектов (продолжение)

Пороговая маска сама по себе не содержит понятия «автомобиль». Она лишь показывает изменение изображения. Для распознавания транспортного средства необходимы дополнительные признаки, классификатор либо обученная нейросетевая модель.

```
% MATLAB: базовая маска разности двух кадров
A = zeros(120,200,'uint8');
B = A;
B(45:75,80:125) = 180; % синтетический объект
threshold = 40;
mask = abs(double(B)-double(A)) > threshold;
imagesc(mask); axis image; colormap(gray);
title('Бинарная маска движения');
```

# 6 Морфологическая обработка и связные компоненты

После пороговой обработки маска часто содержит одиночные шумовые пиксели, отверстия внутри объекта и разрывы границ. Для очистки применяют морфологические операции: эрозию, дилатацию, открытие и закрытие. Их параметры зависят от масштаба автомобилей на изображении.

Открытие удаляет небольшие изолированные области, а закрытие заполняет малые промежутки. Чрезмерное закрытие может объединить два соседних автомобиля в один объект, поэтому увеличение структурирующего элемента не всегда улучшает результат.

**Связная компонента —** группа соседних пикселей бинарной маски. Для каждой компоненты можно вычислить площадь, ограничивающий прямоугольник и центр масс. Малые области отбрасывают по порогу площади, однако его следует выбирать с учётом перспективы.

В MATLAB функции bwareaopen, imopen, imclose и regionprops относятся к инструментам обработки изображений; их доступность зависит от установленных продуктов. В учебном проекте необходимо отдельно указать требуемый Image Processing Toolbox.

# 6 Морфологическая обработка и связные компоненты (продолжение)

Объект, выделенный как одна связная компонента, не обязательно является автомобилем. Тень может увеличить область, а перекрытие двух машин — объединить их. Поэтому связные компоненты являются промежуточным этапом обработки, а не гарантированным подсчётом.

```
% MATLAB: пример требует Image Processing Toolbox
BW = false(100,160);
BW(35:60,50:90) = true;
BW(10,10) = true; % шум
BWclean = bwareaopen(BW,20);
stats = regionprops(BWclean,'Area','Centroid','BoundingBox');
fprintf('Компонент после очистки: %d\n',numel(stats));
```

# 7 Отслеживание объектов и пересечение контрольной линии

Для подсчёта уникальных машин каждой обнаруженной области назначают идентификатор, который должен сохраняться между соседними кадрами. Простая схема сопоставляет центры объектов по минимальному расстоянию; более сложные методы учитывают скорость, направление и прогноз положения.

Если координата центра объекта в кадре t−1 находится по одну сторону контрольной линии, а в кадре t — по другую, регистрируется событие пересечения. Для горизонтальной линии y=y0 условие можно записать как (y_{t−1}<y0) и (y_t≥y0) для выбранного направления движения.

Каждый идентификатор должен засчитываться не более одного раза на одной линии. Иначе колебание центра около границы приведёт к повторному счёту. Для борьбы с дрожанием применяют зону гистерезиса и хранение уже засчитанных ID.

При перекрытии машин, резких манёврах и пропусках кадров простой алгоритм ближайшего центра может перепутать идентификаторы. Для реального трафика используют более устойчивые трекеры, например методы прогнозирования движения и сопоставления детекций.

7

# Отслеживание объектов и пересечение контрольной линии (продолжение)

Для двусторонней дороги линии подсчёта и правила направления задают отдельно по полосам. Число пересечений линии — наблюдаемая метрика, а не автоматическая оценка общего числа автомобилей на всей трассе.

```
% MATLAB: учебная логика пересечения линии
lineY = 60;
yPrevious = [45 70 55];
yCurrent  = [65 75 62];
crossed = (yPrevious < lineY) & (yCurrent >= lineY);
fprintf('Пересечений вниз: %d\n',sum(crossed));
```

8

# Транспортные показатели и математические формулы

Интенсивность транспортного потока q определяется количеством транспортных средств N, прошедших через контрольное сечение за время Δt: q=N/Δt. Если Δt выражено в часах, единица q — автомобиль/час. Например, 18 машин за 60 секунд соответствуют 1080 автомобилям/час при условии репрезентативности интервала.

Плотность k характеризует число автомобилей на единицу длины участка: k=N_участка/L, где L — длина участка в километрах. Оценить k по одной линии подсчёта без дополнительных пространственных данных нельзя.

Средняя пространственная скорость v и интенсивность q связаны фундаментальным соотношением q≈k·v для согласованного стационарного потока и совместимых определений показателей. Применение этой формулы к кратковременным нестационарным данным требует осторожности.

Для автомобиля, проходящего известное расстояние S между двумя контрольными линиями за время Δt, скорость оценивают как v=S/Δt. Перевод из м/с в км/ч: v_km/h=3,6·v_m/s. Без метрической калибровки расстояния результат не является физической скоростью.

8

# Транспортные показатели и математические формулы (продолжение)

Занятость контрольной зоны можно оценивать долей времени, в течение которого она занята обнаруженным транспортным средством: O=(T_занято/T_наблюдения)·100 %. Эта величина зависит от определения зоны и алгоритма обнаружения.

# 9 MATLAB-прототип для анализа последовательности кадров

Разработку рекомендуется начинать с воспроизводимого синтетического примера, в котором объект перемещается по известной траектории. Это позволяет проверить логику пересечения линии и расчёт интенсивности без ошибок реального детектора.

**После проверки алгоритм адаптируют к видеозаписи:** считывают кадры, получают центры обнаруженных объектов, сопоставляют их между кадрами и регистрируют пересечения. Функции VideoReader и средства обработки изображений используются в зависимости от формата видео и установленных инструментов.

При обработке видео нельзя просто считать количество объектов на каждом кадре и суммировать его. Нужно хранить состояние треков и признак того, был ли объект ранее засчитан.

Ниже приведён автономный пример MATLAB, который моделирует координаты трёх автомобилей и проверяет пересечение линии. Это проверка алгоритма подсчёта, а не детектор реальных автомобилей.

Для инженерного отчёта следует указать, какие части программы выполняются на синтетических данных, а какие проверены на реальном видео.

9

# MATLAB-прототип для анализа последовательности кадров (продолжение)

```
% MATLAB: модель движения трёх автомобилей
clear; clc;
lineY = 60;
tracks = [20 35 50 65 80; ...
          10 20 30 40 50; ...
          40 48 56 64 72];
count = 0;
for id = 1:size(tracks,1)
    y = tracks(id,:);
    event = any(y(1:end-1)<lineY & y(2:end)>=lineY);
    if event
        count = count + 1;
    end
end
windowSeconds = 60;
flowPerHour = count/windowSeconds*3600;
fprintf('Машин: %d; поток: %.0f авт/ч\n',count,flowPerHour);
```

# 10 Передача агрегированных показателей в ThingSpeak

Для публикации результатов достаточно передавать число автомобилей за интервал, расчётную интенсивность и показатель качества обработки. Например, Field 1 — N за 60 секунд, Field 2 — q в автомобилях/час, Field 3 — доля успешно обработанных кадров.

На Raspberry Pi вычисление метрик может выполняться непрерывно, тогда как публикация — один раз по завершении окна наблюдения. Перед передачей нужно проверить действующие ограничения частоты обновления ThingSpeak и доступные возможности аккаунта.

Для записи используют HTTPS API или функцию thingSpeakWrite при наличии соответствующей поддержки MATLAB. Секретный ключ записи нельзя включать в публичный код; необходимо проверять ответ сервера и вести локальный журнал.

Если сеть недоступна, показатели сохраняют с исходными временными метками и отправляют позднее допустимым способом. Важно не смешивать время измерения с временем поступления данных на сервер.

ThingSpeak позволяет строить графики изменения потока по времени и проводить MATLAB-анализ. Однако облачная задержка делает эту схему подходящей прежде всего для мониторинга и исследований, а не для жёсткого оперативного управления транспортом.

10

# Передача агрегированных показателей в ThingSpeak (продолжение)

```
% MATLAB: шаблон публикации (не выполнять без ключа)
channelID = 123456;  % собственный канал
writeKey = 'YOUR_WRITE_API_KEY';
count = 18;
q = count/60*3600;
% thingSpeakWrite(channelID,[count q], ...
%     'Fields',[1 2],'WriteKey',writeKey);
fprintf('Подготовлено: %d машин, %.0f авт/ч\n',count,q);
```

# 11 Погрешности, валидация и вычислительные ограничения

Качество подсчёта проверяют по вручную размеченному эталонному видео. Для каждого интервала сравнивают истинное число автомобилей N_ref и алгоритмическое N_alg. Абсолютная ошибка E=|N_alg−N_ref|; относительная ошибка E_rel=|N_alg−N_ref|/N_ref·100 % при N_ref>0.

Для оценки обнаружения объектов применяют precision=TP/(TP+FP) и recall=TP/(TP+FN), где TP — корректные обнаружения, FP — ложные, FN — пропущенные. Эти показатели относятся к согласованной процедуре сопоставления предсказаний с разметкой.

При анализе событий пересечения следует отдельно учитывать повторный счёт, пропуск автомобиля и неправильное направление. Для корректной оценки нужны размеченные события, а не только количество объектов на отдельных кадрах.

Производительность Raspberry Pi характеризуют обработанными кадрами в секунду, загрузкой CPU, потреблением памяти и задержкой публикации. Если обработка медленнее поступления кадров, следует определить политику пропуска или буферизации кадров.

**Важна репрезентативность:** результаты, полученные днём в сухую погоду, не доказывают работоспособность ночью, при дожде или плотном заторе. Для научной оценки требуется набор сценариев и несколько повторных испытаний.

# 12 Приватность, безопасность и план эксперимента

Видеозаписи дорожного движения могут содержать лица, государственные регистрационные знаки и другие идентифицирующие признаки. Для учебного проекта предпочтительно использовать разрешённые записи, минимизировать хранение исходного видео и передавать в облако только агрегированные показатели.

Необходимо соблюдать требования законодательства и правил владельца дорожной инфраструктуры. Установка камеры на реальной трассе не должна создавать помех движению или нарушать безопасность людей.

**Этап 1 —** определить метрику и контрольную линию. Этап 2 — получить разрешённое видео и зафиксировать параметры съёмки. Этап 3 — реализовать обнаружение движения. Этап 4 — очистить маски и выделить компоненты. Этап 5 — добавить отслеживание и подсчёт пересечений.

**Этап 6 —** вручную разметить контрольный фрагмент, оценить ошибки и производительность. Этап 7 — настроить ThingSpeak и отправку агрегированных данных. Этап 8 — проверить отказ сети и восстановление передачи. Этап 9 — подготовить отчёт с ограничениями метода.

Отчёт должен включать схему системы, описание камеры и сцены, математические формулы, исходный код, графики потока, эталонный подсчёт, метрики ошибок и вывод о применимости выбранного подхода.

# 3 Заключение

В лекции рассмотрены основные этапы создания системы анализа дорожного трафика с использованием Raspberry Pi, веб-камеры, MATLAB и ThingSpeak: получение кадров, выделение движения, морфологическая обработка, отслеживание объектов, подсчёт пересечений контрольной линии и расчёт интенсивности. Подчёркнуто, что точность реальной системы зависит от калибровки, условий съёмки, качества трекинга и независимой проверки по размеченным данным. ThingSpeak используется для передачи агрегированных показателей, а не полного видеопотока.

# Контрольные вопросы (1/2)

1. Какие задачи решает анализ дорожного потока?
2. Почему число обнаружений на кадрах не равно числу автомобилей?
3. Какие компоненты входят в архитектуру Raspberry Pi — MATLAB — ThingSpeak?
4. Почему в облако предпочтительно передавать агрегированные показатели?
5. Какие параметры камеры влияют на качество анализа?
6. Что такое область интереса ROI?
7. Как выполняется преобразование RGB в оттенки серого?
8. Как вычисляется разность кадра и фона?
9. Что означает порог τ в бинарной маске движения?
10. Для чего используется адаптивная модель фона?
11. Какие проблемы создают тени и изменение освещения?
12. Что такое морфологическое открытие и закрытие?
13. Для чего вычисляются связные компоненты?
14. Почему объединение двух машин в одну компоненту является ошибкой?

# Контрольные вопросы (2/2)

15. Что такое трек объекта?

16. Как определяется пересечение контрольной линии?

17. Как избежать повторного подсчёта одного автомобиля?

18. Как вычисляется интенсивность в автомобилях/час?

19. Почему нельзя оценить физическую скорость без калибровки?

20. Что характеризует занятость контрольной зоны?

21. Какие данные рекомендуется публиковать в ThingSpeak?

22. Как рассчитываются precision и recall?

23. Как измерить ошибку подсчёта автомобилей?

24. Какие ограничения Raspberry Pi влияют на обработку видео?

25. Какие требования приватности важны при видеонаблюдении дороги?

# Д Задания для самостоятельной работы

1. Постройте структурную схему видеомониторинга дорожного потока с Raspberry Pi, MATLAB и ThingSpeak.

2. Создайте в MATLAB два синтетических кадра и реализуйте бинарную маску движения с тремя порогами.

3. Смоделируйте координаты пяти автомобилей и подсчитайте пересечения горизонтальной линии без повторного счёта.

4. Вычислите интенсивность для 12, 20 и 35 автомобилей за 60 секунд и сравните результаты.

5. Разработайте структуру канала ThingSpeak для публикации числа машин, интенсивности и показателя качества.

6. Составьте план эксперимента с ручной разметкой, расчётом ошибок и проверкой работы при потере сети.

# Рекомендуемые источники

- Силлабус дисциплины «Проект MATLAB IoT», тема лекции №13.
- MathWorks. MATLAB Documentation — https://www.mathworks.com/help/matlab/
- MathWorks. Image Processing Toolbox Documentation — https://www.mathworks.com/help/images/
- MathWorks. Computer Vision Toolbox Documentation — https://www.mathworks.com/help/vision/
- MathWorks. ThingSpeak Documentation — https://www.mathworks.com/help/thingspeak/
- Raspberry Pi Documentation — https://www.raspberrypi.com/documentation/