به نام خدا

ساینا خورشیدزاده سکشن چهارشنبه ۱۳ . ولی برای پروژه دوشنبه ۱۳ دستیار صوتی هوشمند

کد دانشجویی:۴۰۱۱۶۳۴۱۰۵۴۴۱۲

percept			
Current State/ حالت فعلی	Input / دریافتی .ورودی	Next State/حالت بعدی	Action
Idle	Voice command	Listening	Start listening
Listening	End of command	Processing	Process the command
Processing	Successful process	Responding	Play voice response
Processing	Processing error	Error	Display error message
Responding	End of response	Idle	Return to idle state
Responding	Needs confirmation	Waiting for Confirmation	Ask for user confirmation
Waiting for Confirmation	User confirms	Executing Command	Execute the command
Executing Command	Command executed	Idle	Return to idle state
Idle	Update available	Updating	Start updating
Updating	Update complete	Idle	Return to idle state
Idle	User interaction	Learning	Start learning
Learning	Learning complete	Idle	Return to idle state
Idle	No interaction for a while	Sleeping	Enter sleep mode
Sleeping	Wake word	Idle	Wake up
Idle	Notification trigger	Notification	Send notification
Notification	Notification sent	Idle	Return to idle state
Idle	Device interaction trigger	Interacting with Other Devices	Start device interaction
Interacting with Other Devices	Interaction complete	Idle	Return to idle state
Listening	Record command	Recording	Start recording
Recording	End of recording	Processing	Process the recording
Listening	Translation command	Translating	Start translating
Translating	Translation complete	Responding	Provide translated response
Idle	Obstacle detected	Obstacle Avoidance	Start obstacle avoidance
Obstacle Avoidance	Obstacle avoided	Idle	Return to idle state
Obstacle Avoidance	Obstacle not avoided	Error	Display error message
Idle	Lane departure detected	Lane Keeping	Start lane keeping
Lane Keeping	Lane maintained	Idle	Return to idle state
Lane Keeping	Lane not maintained	Error	Display error message
Idle	Traffic sign detected	Traffic Sign Recognition	Start traffic sign recognition
Traffic Sign Recognition	Sign recognized	Responding	Provide traffic sign information
Traffic Sign Recognition	Sign not recognized	Error	Display error message
Idle	Speed limit exceeded	Speed Control	Start speed control
Speed Control	Speed within limit	Idle	Return to idle state
Speed Control	Speed not within limit	Error	Display error message
Idle	Collision risk detected	Collision Avoidance	Start collision avoidance
Collision Avoidance	Collision avoided	Idle	Return to idle state
Collision Avoidance	Collision not avoided	Error	Display error message
Idle	Destination reached	Parking	Start parking
Parking	Parked successfully	Idle	Return to idle state
Parking	Parking failed	Error	Display error message
Idle	Low fuel detected	Fuel Management	Start fuel management

Fuel Management	Fuel managed	Idle	Return to idle state
Fuel Management	Fuel not managed	Error	Display error message
Idle	Weather change detected	Weather Adjustment	Start weather adjustment
Weather Adjustment	Weather adjusted	Idle	Return to idle state
Weather Adjustment	Weather not adjusted	Error	Display error message
Idle	Route recalculation needed	Route Recalculation	Start route recalculation
Route Recalculation	Route recalculated	Idle	Return to idle state
Route Recalculation	Route not recalculated	Error	Display error message
Idle	Passenger request detected	Passenger Interaction	Start passenger interaction
Passenger Interaction	Request fulfilled	Idle	Return to idle state
Passenger Interaction	Request not fulfilled	Error	Display error message
Idle	Maintenance needed	Maintenance	Start maintenance
Maintenance	Maintenance completed	Idle	Return to idle state
Maintenance	Maintenance not completed	Error	Display error message
Idle	Emergency detected	Emergency Handling	Start emergency handling
Emergency Handling	Emergency handled	Idle	Return to idle state
Emergency Handling	Emergency not handled	Error	Display error message

حالات دستیار صوتی هوشمند :

Idle:**دستیار در حالت آماده به کار است(بیکار)**

Listening:**دستیار در حال گوش دادن به فرمان صوتی کاربر است**

Processing:**دستیار در حال پردازش فرمان صوتی است**

Responding:**دستیار در حال پخش پاسخ صوتی است**

Error:**دستیار با خطا مواجه شده است**

Waiting for Confirmation:**دستیار منتظر تأیید کاربر برای انجام یک عمل است**

Executing Command:**دستیار در حال اجرای فرمان کاربر است**

Updating:**دستیار در حال به روزرسانی نرم افزار یا دیتا است**

Learning:**دستیار در حال یادگیری از تعاملات کاربر است**

Sleeping:**دستیار در حالت خواب است و باید بیدار شود**

Notification:**دستیار در حال ارسال نوتیفیکیشن به کاربر است**

Interacting with Other Devices:**دستیار در حال تعامل با دستگاههای دیگر است**

Recording:**دستیار در حال ضبط صدای کاربر است**

Translating:**دستیار در حال ترجمه زبان کاربر است**

Obstacle Avoidance: **دستیار در حال اجتناب از موانع است**

Lane Keeping: دستیار در حال حفظ مسیر است

Traffic Sign Recognition: دستیار در حال تشخیص علائم ترافیکی است

Speed Control: دستیار در حال کنترل سرعت است

Collision Avoidance: دستیار در حال اجتناب از تصادف است

Parking: دستیار در حال پارک کردن است

Fuel Management: دستیار در حال مدیریت سوخت است

Weather Adjustment: دستیار در حال تنظیم شرایط آب و هوایی است

Route Recalculation: دستیار در حال محاسبه مجدد مسیر است

Passenger Interaction: دستیار در حال تعامل با مسافران است

Maintenance: دستیار در حال انجام تعمیرات است

Emergency Handling:دستیار در حال مدیریت وضعیت اضطراری است

کد مربوط با حالت عامل مبتنی بر سودمندی :

```
import heapq

#تعریف گراف حالتها و هزینهها
graph = {
"Idle": {"Listening": 1, "Updating": 2, "Obstacle Avoidance": 3},
"Listening": {"Processing": 1},
"Processing": {"Responding": 1, "Error": 5},
"Responding": {"Idle": 1, "Waiting for Confirmation": 2},
"Waiting for Confirmation": {"Executing Command": 1, "Idle": 2},
"Executing Command": {"Idle": 1},
"Updating": {"Idle": 1},
"Learning": {"Idle": 1},
"Sleeping": {"Idle": 1},
"Notification": {"Idle": 1},
"Interacting with Other Devices": {"Idle": 1},
"Recording": {"Processing": 1},
"Translating": {"Responding": 1},
"Obstacle Avoidance": {"Idle": 1, "Error": 5},
"Lane Keeping": {"Idle": 1, "Error": 5},
"Traffic Sign Recognition": {"Responding": 1, "Error": 5},
"Speed Control": {"Idle": 1, "Error": 5},
"Collision Avoidance": {"Idle": 1, "Error": 5},
"Parking": {"Idle": 1, "Error": 5},
"Fuel Management": {"Idle": 1, "Error": 5},
"Weather Adjustment": {"Idle": 1, "Error": 5},
"Route Recalculation": {"Idle": 1, "Error": 5},
"Passenger Interaction": {"Idle": 1, "Error": 5},
"Maintenance": {"Idle": 1, "Error": 5},
"Emergency Handling": {"Idle": 1, "Error": 5}
}

#تعریف امتیازهای سودمندی برای هر حالت
utility_scores = {
"Idle": 1,
"Listening": 2,
"Processing": 3,
"Responding": 2,
"Error": -1,
"Waiting for Confirmation": 1,
"Executing Command": 4,
"Updating": 1,
"Learning": 2,
"Sleeping": 0,
"Notification": 2,
"Interacting with Other Devices": 3,
"Recording": 2,
"Translating": 3,
"Obstacle Avoidance": 5,
"Lane Keeping": 4,
"Traffic Sign Recognition": 4,
"Speed Control": 4,
"Collision Avoidance": 5,
"Parking": 3,
"Fuel Management": 3,
"Weather Adjustment": 3,
"Route Recalculation": 4,
"Passenger Interaction": 2,
"Maintenance": 1,
"Emergency Handling": 5
}

def a_star_search(start, goal):
    open_list = []
    heapq.heappush(open_list, (0, start))
    came_from = {}
    cost_so_far = {start: 0}

    while open_list:
        current_priority, current_state = heapq.heappop(open_list)

        if current_state == goal:
            break

        for next_state in graph[current_state]:
            new_cost = cost_so_far[current_state] + graph[current_state][next_state]
            if next_state not in cost_so_far or new_cost < cost_so_far[next_state]:
                cost_so_far[next_state] = new_cost
                priority = new_cost - utility_scores[next_state]
                heapq.heappush(open_list, (priority, next_state))
                came_from[next_state] = current_state

    return reconstruct_path(came_from, start, goal)

def reconstruct_path(came_from, start, goal):
    current = goal
    path = []
    while current != start:
        path.append(current)
        current = came_from[current]
    path.append(start)
    path.reverse()
    return path

def select_best_state(current_state, possible_states):
    best_state = current_state
    highest_utility = utility_scores[current_state]

    for state in possible_states:
        if utility_scores[state] > highest_utility:
            best_state = state
            highest_utility = utility_scores[state]

    return best_state

#تابع برای دریافت ورودیهای کاربر
def get_user_input():
    user_input = input("Enter your command: ")
    return user_input
```

```
def manage_states():
    current_state = "Idle"
    goal_state = "Executing Command"

    while current_state != goal_state:
        print(f"Current State: {current_state}")

#دریافت ورودی کاربر
        user_input = get_user_input()

#تعریف حالتهای ممکن بر اساس وضعیت فعلی و ورودی کاربر
        if current_state == "Idle":
            if user_input == "listen":
                possible_states = ["Listening"]
            elif user_input == "update":
                possible_states = ["Updating"]
            elif user_input == "avoid obstacle":
                possible_states = ["Obstacle Avoidance"]
            else:
                possible_states = ["Error"]
        elif current_state == "Listening":
            if user_input == "process":
                    possible_states = ["Processing"]
            else:
                possible_states = ["Error"]
        elif current_state == "Processing":
            if user_input == "respond":
                possible_states = ["Responding"]
            elif user_input == "error":
                possible_states = ["Error"]
            else:
```

```
        possible_states = ["Error"]
elif current_state == "Responding":
    if user_input == "idle":
        possible_states = ["Idle"]
    elif user_input == "wait":
        possible_states = ["Waiting for Confirmation"]
    else:
        possible_states = ["Error"]
elif current_state == "Waiting for Confirmation":
    if user_input == "execute":
        possible_states = ["Executing Command"]
    elif user_input == "idle":
        possible_states = ["Idle"]
    else:
        possible_states = ["Error"]
elif current_state == "Executing Command":
    if user_input == "idle":
        possible_states = ["Idle"]
    else:
        possible_states = ["Error"]
elif current_state == "Updating":
    if user_input == "idle":
        possible_states = ["Idle"]
    else:
        possible_states = ["Error"]
elif current_state == "Learning":
    if user_input == "idle":
        possible_states = ["Idle"]
    else:
        possible_states = ["Error"]
elif current_state == "Sleeping":
    if user_input == "idle":
        possible_states = ["Idle"]
    else:
        possible_states = ["Error"]
elif current_state == "Notification":
    if user_input == "idle":
            possible_states = ["Idle"]
    else:
        possible_states = ["Error"]
elif current_state == "Interacting with Other Devices":
    possible_states = ["Idle"]
elif current_state == "Recording":
    possible_states = ["Processing"]
elif current_state == "Translating":
    possible_states = ["Responding"]
elif current_state == "Obstacle Avoidance":
    possible_states = ["Idle", "Error"]
elif current_state == "Lane Keeping":
    possible_states = ["Idle", "Error"]
elif current_state == "Traffic Sign Recognition":
    possible_states = ["Responding", "Error"]
elif current_state == "Speed Control":
    possible_states = ["Idle", "Error"]
elif current_state == "Collision Avoidance":
    possible_states = ["Idle", "Error"]
elif current_state == "Parking":
    possible_states = ["Idle", "Error"]
elif current_state == "Fuel Management":
    possible_states = ["Idle", "Error"]
elif current_state == "Weather Adjustment":
    possible_states = ["Idle", "Error"]
elif current_state == "Route Recalculation":
    possible_states = ["Idle", "Error"]
elif current_state == "Passenger Interaction":
    possible_states = ["Idle", "Error"]
elif current_state == "Maintenance":
    possible_states = ["Idle", "Error"]
elif current_state == "Emergency Handling":
    possible_states = ["Idle", "Error"]
else:
    possible_states = ["Error"]

#انتخاب بهترین حالت بعدی با استفاده از الگوریتم A*
next_state = select_best_state(current_state, possible_states)
print(f"Next State: {next_state}")

#بهروزرسانی حالت فعلی
current_state = next_state

#شبیهسازی یک وقفه برای مشاهده تغییر حالتها
import time
time.sleep(1)

#اجرای تابع مدیریت حالتها
manage_states()
```