Questions?

- My tasks:
  - IDR sign-up
  - grade Lab 3

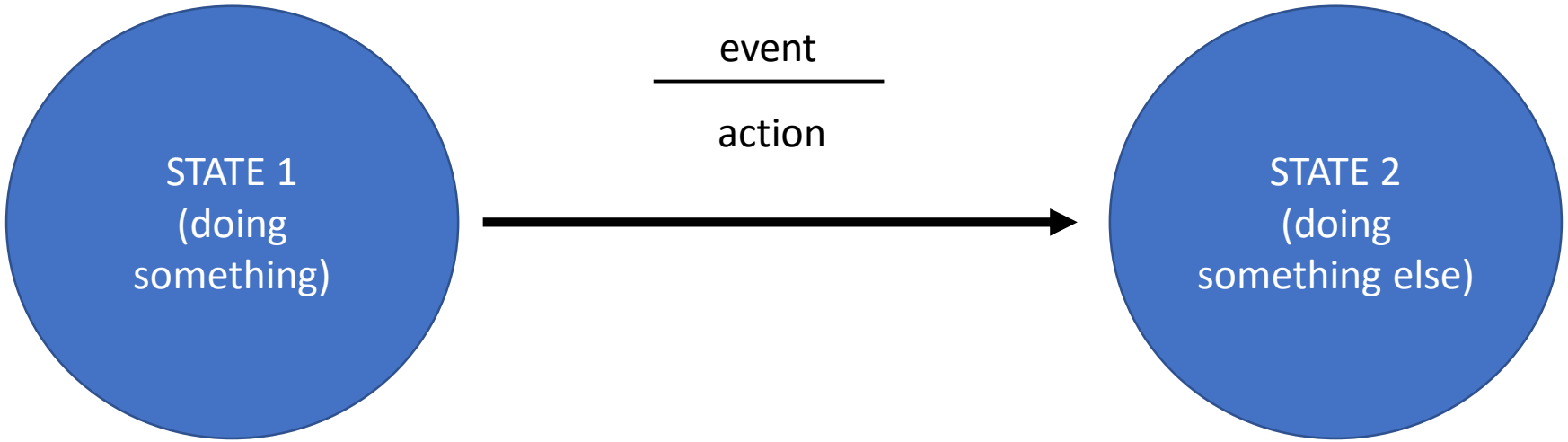
- Your tasks
  - Home-alarm prep: next Tuesday
  - Home-alarm system: next Friday
  - Prep for IDR
  - **Quiz today!**
  - Lab 4 post-lab: Tuesday
  - HW 4.1: Tuesday

State machines

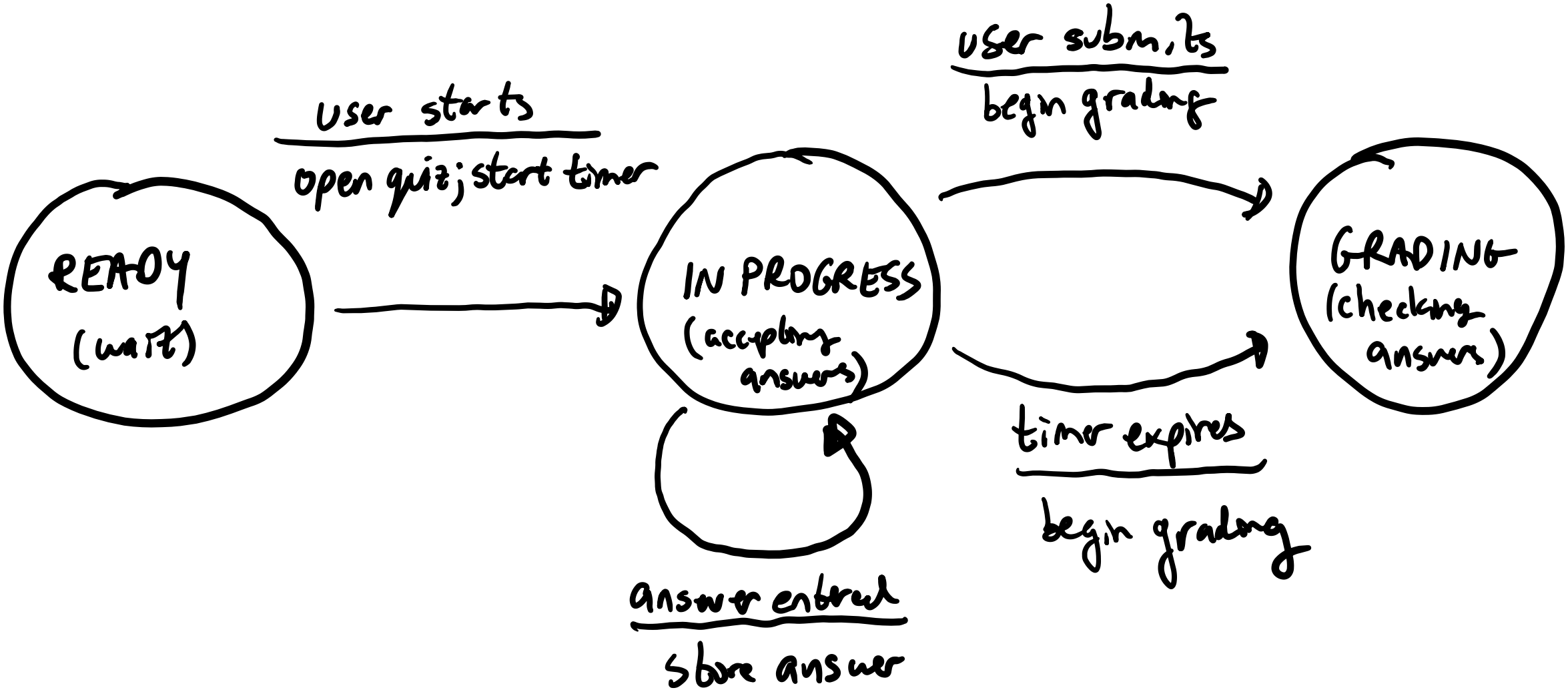
Asynchronous events

- The value of a state machine is that they are useful for detecting and handling *asynchronous events*
  - While driving, detect the presence of a building or roadblock
  - While pivoting a turret, detect an IR beacon
  - While driving, detect an intersection and decide which way to go

Anatomy of a State Transition Diagram



Taking a Quiz



How might you write this in code?

- We'll use the checker/handler framework from yesterday:

```
while(1)
{
  if(SomeEvent() == true) HandleThatEvent();
  if(SomeOtherEvent() == true) HandleThatOtherEvent();
  if(YetAnotherEvent()) HandleThatOtherOtherEvent();
}
```

First, define the states in a readable format

```
enum QUIZ_STATE {READY, IN_PROGRESS, GRADING, CLOSED};

QUIZ_STATE quizState = READY;
```

Define the checker/handler format

```
enum QUIZ_STATE {READY, IN_PROGRESS, GRADING, CLOSED};
QUIZ_STATE quizState = READY;

while(1)
{
  if(openQuizButton.Pressed()) HandleOpenQuizButton();
  .
  .
  .
}
```

Then define the handler function

```
enum QUIZ_STATE {READY, IN_PROGRESS, GRADING, CLOSED};
QUIZ_STATE quizState = READY;

void HandleOpenQuizButton(void)
{
  if(quizState == READY)
  {
    quiz.Open();
    timer.Start();
    quizState = IN_PROGRESS;
  }
}
```

What goes here?

```
enum QUIZ_STATE {READY, IN_PROGRESS, GRADING, CLOSED};
QUIZ_STATE quizState = READY;

void HandleSubmitQuizButton(void)
{
  if(quizState==IN_PROGRESS) {
    quiz.StartGrading();
    quizState=GRADING;
  }
}
```

What goes here?

```
enum QUIZ_STATE {READY, IN_PROGRESS, GRADING, CLOSED};
QUIZ_STATE quizState = READY;

void HandleSubmitQuizButton(void)
{
  if(quizState == IN_PROGRESS)
  {
    quiz.StartGrading();
    quizState = GRADING;
  }
}
```

What might a state machine look like?

- You are designing a robot to collect items lying around the lab and put them in a collection bin
- Items can be found most anywhere (have you seen the lab?) and can be detected with a camera
- The camera can also detect the location of the collection bin, which can be seen from every point in the room – you only need to spin in place to locate it
- Your robot has a means of sensing distance from an object/collection bin
- Your robot has an exquisitely constructed grasping mechanism that is good, but not perfect: sometimes it drops items
- Because it’s not perfect, you’ve put a switch in the grasper so your robot knows when an object is present

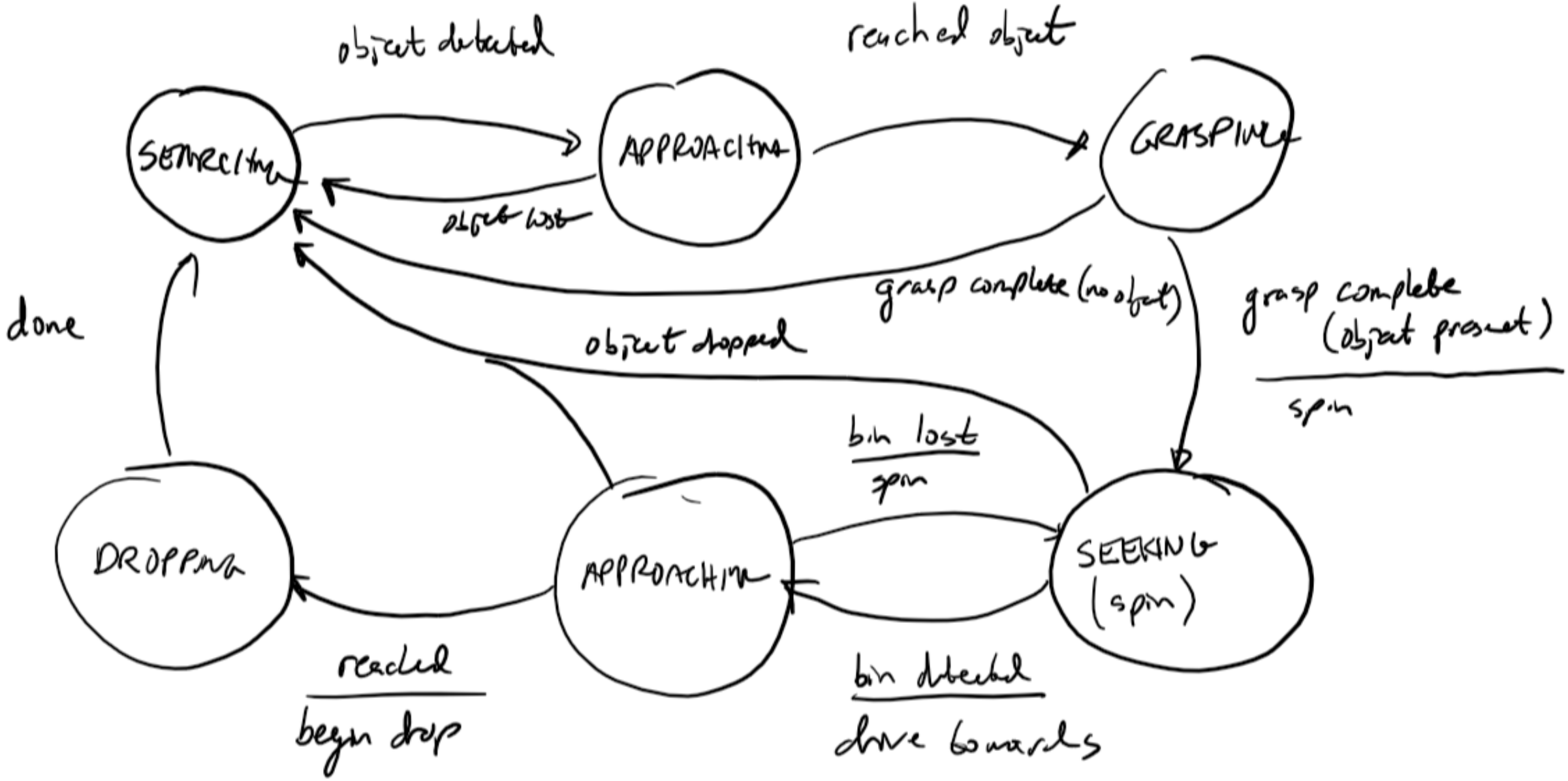
Define the states

Define the transitions

- Start with the successful case

Define the transitions

- Now fill in the other possibilities


object detected
reached object
SEARCHING
APPROACHING
GRASPING
object lost
grasp complete (no object)
grasp complete
(object present)
spin
done
object dropped
bin lost
spin
DROPPING
APPROACHING
SEEKING
(spin)
reached
begin drop
bin detected
drive towards

Terminology