The Role of Synchronic Causal Conditions in Visual Knowledge Learning

Seng-Beng Ho
Institute of High Performance Computing, A*STAR
1 Fusionopolis Way, #16-16 Connexis North, Singapore
hosengbeng@gmail.com

Abstract

We propose a principled approach for the learning of causal conditions from actions and activities taking place in the physical environment through visual input. Causal conditions are the preconditions that must exist before a certain effect can ensue. We propose to consider diachronic and synchronic causal conditions separately for the learning of causal knowledge. Diachronic condition captures the "change" aspect of the causal relationship – what change must be present at a certain time to effect a subsequent change – while the synchronic condition is the "contextual" aspect – what "static" condition must be present to enable the causal relationship involved. This paper focuses on discussing the learning of synchronic causal conditions as well as proposing a principled framework for the learning of causal knowledge including the learning of extended sequences of cause-effect and the encoding of this knowledge in the form of scripts for prediction and problem solving.

1. Introduction

Being able to discern causalities between events is of paramount importance for an intelligent system. Knowing the causalities between events allows the system to make predictions and carry out actions for problem solving. A number of previous research works have investigated methods for learning causalities based on observing temporal correlations through vision [1-3]. Causality dictates a certain temporal order between cause and effect. Having learned the causality, say, between two events, subsequently, the temporal order observed between them can be used to discern the causality involved.

However, Ho and Liausvia [2] pointed out that depending on the kind of sensory information that is available in attempting to discern causalities, in some situations it is difficult to discern causality based on temporal order alone. An example given was observing the sound of a gunshot and the subsequent sound emitted by or visual damage seen on the target. If two guns were fired at two targets, one could observe two consecutive gunshots first, followed by two consecutive sounds emitted by, or two consecutive visual changes occurring on, the targets. In this situation, it is difficult to establish which gun causes the damage on which target.

However, if more information is available from the visual scene, the causalities could be adequately discerned. Basically, the fact is that for a gun to emit a bullet to hit a target, the gun has to be pointing at the target. This piece of knowledge could be encoded as follows:

$$\forall x, y \quad Gun(x) \wedge Relative\text{-}Angle((Barrel(x), Object(y)) = 0, t) \wedge Shoot(x, t) \rightarrow Damage(Object(y), t+\Delta) \qquad (1)$$

which is a predicate logical statement stating that for every *x* that is a *Gun*, if the *Relative-Angle* between the long axis of the gun *Barrel* of *x* and a line joining the gun to *Object(y)* is *0* at time *t* (this is the definition of the *Relative-Angle* predicate) and *x Shoots* (a bullet) at time *t*, then *Object(y)* is *Damaged* at time $t+\Delta$. Then, using this knowledge and by observing which gun is pointing at which object, it can be discerned which gun causes the corresponding damage.

Thus, the availability of extra visual information and the availability of a more detailed physical model of the causal process involved can assist in discerning the causalities involved. In this paper, we expand on the work of Ho and Liausvia [2, 3] and describe a framework in which the above piece of causal knowledge as encoded in Eqn. 1 can be learned. It will be seen that learning causal knowledge such as that encoded in Eqn. 1 involves more than learning the temporal correlation between cause and effect. There is other accompanying sensory and physical information that has to be learned and encoded in certain manners as well.

2. A Framework for Learning and Encoding Causal Knowledge

In this section, a framework for the learning and encoding of causal knowledge is described. This involves

the principled learning of what are termed the *diachronic* and *synchronic* causal conditions, as well as the learning and encoding of sequences of causally connected actions into event scripts [4-8]. The scripts encode causal-spatial-temporal knowledge, similar to the knowledge structures encoded in causal-spatial-temporal AND-OR graphs as investigated by Tu et al. [9] but with extra information learned and encoded that can better represent the causal and physical properties of the objects and processes involved for the use of subsequent processes such as prediction and problem solving. The primary difference between the framework developed here with regards to scripts and that of some previous work on scripts [4-6] is that this previous work does not deal with learning of scripts directly from visual information. Compared with some other previous work that discusses learning of scripts from visual information [7, 8], we present a more complete and principled framework of learning and encoding of causal knowledge into scripts in this paper.

2.1. Diachronic and Synchronic Causal Conditions

Some previous work [7, 8, 10] has shown that it is essential to separate two basic kinds of causal conditions – the diachronic and synchronic causal conditions. We use a simple scenario in Figure 1 to illustrate this idea.

In Figure 1 it is shown that there are 3 objects, A, B, and C in an environment represented by the rectangle. Suppose we ignore the presence and influence of the rectangle and consider just the objects.

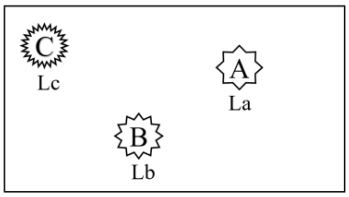

Figure 1: Example used to illustrate the idea of diachronic and synchronic causal conditions.

Consider that C always exists, at location Lc. This is represented as *Exist(C, Lc)*. Consider that A and B do not always exist and A appears at time T and then shortly after that (time Δ later) B appears, and that the locations of A and B are La and Lb respectively. This is known as the *Appear* action. Suppose we recognize a causal relation between the appearance of A and B. The relation is stated as *Appear(A, La, T)→Appear(B, Lb, T+Δ)*. The appearance of A is termed the *diachronic* (*DIA*) causal condition of the causal relationship. Suppose also that we recognize that the existence of C at location Lc is necessary for the causal relationship to exist (i.e., without the presence of C, the appearance of A at La cannot cause the appearance of B at Lb). This is represented as:

$$Exist(C, Lc)\ \textbf{[SYN]} \wedge Appear(A, La, T)\ \textbf{[DIA]} \rightarrow Appear(B, Lb, T+\Delta) \quad (2)$$

Exist(C, Lc) is known as the *synchronic* (*SYN*) causal condition, the absence of which entails the non-existence of the diachronic causal relation. It can be thought of as an "enabling" causal condition – i.e., it enables the diachronic causal relation to take place. The idea of "enabling" causal conditions has been investigated by Abelson in an earlier effort [11]. If we think of it in terms of counterfactual function, then it is like "had it not been there, the diachronic cause would not have given rise to the effect." Its counterfactual causal role is the same as that of the diachronic causal condition – "had the diachronic cause not been there, the effect would not have taken place." In the following discussions we will not explicitly label DIA and SYN for the sake of succinctness but their respective roles will be obvious. The learning mechanisms for the identification of these conditions will be discussed in the next two sections.

2.2. Learning of Diachronic Causal Conditions

Fire and Zhu [1], and Ho and Liausvia [2, 3] have described methods for learning causalities through video observation. In this paper we will extend on the method of Ho and Liausvia [2, 3] and describe below the essentials of the method that are relevant to the discussion in this paper.

Figure 2: (a) 3 events, A, B, and C are observed to occur in sequence. (b) B is sometimes observed not to take place between A and C.

Suppose there are 3 events, A, B, and C, that are observed to occur in sequence as shown in Figure 2(a). At this moment, Ho and Liausvia's method [2, 3] first proposes the two causal relations: A→B and B→C. If B is indeed always present after A and before C, it is deemed that A is an indirect cause of C and hence it is not necessary to encode A→C separately. However, if B is sometimes not observed to take place between A and C, such as shown in Figure 2(b), then the confidence values for A→B and B→C are reduced. B could just be noise, and the A→C relation is then proposed, and the confidence value associated with it will be adjusted accordingly dependent on whether subsequently C always follows A or A always precedes C. The method is related

to the psychological contingency model of psychology [12, 13] in which the causal contingency ΔP between an effect, *e*, and a cause, *i*, is defined as:

$$\Delta P = P(e / i) - P(e / \neg i) \quad (3)$$

which basically says that if an effect is observed in the absence of the cause, it reduces the probability of the causal contingency involved. ($P(e / i)$ is the probability of observing *e* followed by *i*.

While Ho and Liausvia's method [2, 3] takes care of intervening "noisy events" between two events that are supposedly in a causal relationship, in the current discussion we assume that the situation is ideal and there is no intervening noise, so that a pair of events that consistently follow one another are identified to be causally linked.

2.3. Learning of Synchronic Causal Conditions

As discussed in Section 2.1, synchronic causal conditions are the "contextual" conditions that must be present to allow a cause to bring about an effect. The phenomenon of gravity is a good example to illustrate this idea [8, 10] - the release of an object (*Object(x)*) held in one's hand at one time instance *T causes* the falling of the object at the next time instance $T+\Delta$. The diachronic condition is *Release(Object(x), T) and the causal relation* is *Release(Object(x), T)* → *Fall(Object(x), T+Δ)*. This could be learned through the process of learning diachronic conditions described above. However, there are other parameters associated with *Object(x)*, such as its location. The process might begin with a first experience of the *Release*→*Fall* causality/phenomenon at a specific location, say, *X1*, at *T1*. The agent observing this will first encode a *specific* rule consisting of a specific synchronic condition such as:

At(Object(x), X1, T1) ⋀ *Release(Object(x), T1)*
→ *Fall(Object(x), T1+Δ)* (4)

Then, on a second observation at another location, *X2*, another rule is established:

At(Object(x), X2, T2) ⋀ *Release(Object(x), T2)*
→ *Fall(Object(x), T2+Δ)* (5)

Because the *X* parameter values in the *At(Object(x), X)* synchronic condition for both instances are different (one is *X1*, and the other is *X2*), through a process of inductive generalization [7, 8, 10], these two rules could be generalized and combined into a general rule:

At(Object(x)), ANYWHERE, ANYTIME)
⋀ *Release(Object(x), SAME ANYTIME)*
→ *Fall(Object(x), SAME ANYTIME + Δ)* (6)

"*SAME ANYTIME*" means it is the same "anytime" as the one in the first line of the equation. This would be generalizing based on two instances termed "dual instance generalization." [7, 8, 10] If, on the other hand, the values of *X* are the same in the two instances that occurred at different times *T1* and *T2*, say *SX*, then that value is kept as a "must have" specific value for *X* in a combined rule, with the time being generalized to *ANYTIME*:

At(Object(x)), SX, ANYTIME)
⋀ *Release(Object(x), SAME ANYTIME)*
→ *Fall(Object(x), SAME ANYTIME + Δ)* (7)

Depending on a parameter called "desperation" which captures the desperation on the part of the agent involved in using a causal rule such as that above to solve problems, the number of instances observed before generalization could be more or fewer, as discussed in Ho [8].

Ho [8, 10] also discusses backing up from over generalization. For example, the gravity example above (Eqn. 6) of expecting that releasing an object anywhere will cause the object to fall may become invalid in certain orbital or outer space environment. Thus, a more specific rule may be formed:

At(Object(x)), ANYWHERE ON EARTH, ANYTIME)
⋀ *Release(Object(x), SAME ANYTIME)*
→ *Fall(Object(x), SAME ANYTIME + Δ)* (8)

This process is called *retroactive restoration* of the earlier more specific conditions (i.e., *X1* and *X2* are actually all locations on earth) [8, 10]. It can also be formulated as an exception – e.g., the rule is applicable *ANYWHERE except CERTAIN OUTER SPACE LOCATIONS*, etc.

The process can be summarized as follows:

1. Inductively generalize over values of parameters in the diachronic or synchronic causal conditions that are different in different instances, using a *desperation* parameter to control how many instances to consider before executing generalization.
2. Retain the values of the parameters that are the same over instances.
3. Execute *retroactive restoration* if necessary and create exceptional conditions for general rules.

2.4. Learning of Event Scripts

An event typically consists of a number of sub-actions.

These actions are typically causally connected and can be chained and learned in the form of a "script" [4-8]. Typically, a script encodes a starting state, a sequence of actions, and an ending state [4, 7, 8]. The gun-shooting-bullet-damaging-object event discussed in Section 1 is an example of a script consisting of a number of causally linked sub-actions. In general, there are both diachronic and synchronic causal conditions to be learned. Let us now combine the devices described above to illustrate how an event script can be learned through visual observation.

Figure 3(a) illustrates the earlier mentioned gun-shooting-bullet-damaging-object event in Section 1 with typical visual parameters such as those that may be supplied by a computer visual system. These parameters include the absolute location of the gun (represented, say, by its centroid), AL(Gun(x), X), the absolute location of the target object, AL(Object(y), Y), the absolute angle of the gun barrel's long axis in the environment, AA(Barrel(Gun(x)), A), the relative distance between the gun and the target object, RD(Gun(x), Object(y), D), the relative angle between the long axis of the gun and the line joining the gun's centroid to the target object, RA(Barrel(Gun(x)), Object(y), A), etc. For simplicity, we assume that the target object is a point object.

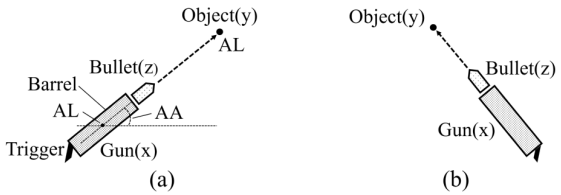

Figure 3: (a) One instance of a Gun(x) at a certain location with a certain orientation, and a target Object(y) at a certain location. (b) Another instance of Gun(x) and Object(y).

Assuming that the bullet (Bullet(z)) is moving slowly enough to be observed, a sequence of actions and the contextual states of Figure 3(a) would be as shown in Figure 4. Figure 4 is the Gun-Shooting-and-Damaging-Object (GSDO) event script learned and captured from the visual environment of the event of Figure 3(a), each line in the script representing the state of the world at that particular time step, starting from an arbitrary time T1. It is a SPECIFIC GSDO script because it is learned from a specific instance of the event, and the various parameters have specific values. One can think of the first line of the script as a "starting" state and the last line as an "outcome" state.

In Figure 4, the time steps are shown on the left as T1, T2, T3, … For this time variable as well as other variables in the figure, such as the location variable X, the number next to the variable represents the value of the variable. I.e., T1 means "T=1," or X6 means "X=6," etc. This notation is used for the sake of brevity. Therefore, the time changes from T1 to T2, T2 to T3, etc. represent elemental time step changes. For simplicity, we use one value "X" to represent both the usual "X" and "Y" co-ordinates in a 2D environment. AL, AA, RD, and RA have the meanings as defined above. Appear, Disappear, Contact, Damage, and Press-Trigger have the usual meanings in English. After Bullet(z) Appears after the Press-Trigger event, it moves toward the target Object(y) and its absolute locations, AL, changes elementally (increasing by one unit for every unit time step) accordingly. The initial synchronic condition (INIT SYN COND) is replicated in every time step.

T1: AL(Gun(x), X1), AL(Object(y), X14), AA(Barrel(Gun(x)), A45),
RD(Gun(x), Object(y), D6), RA(Barrel(Gun(x)), Object(y), 0)
T2: Press-Trigger(Gun(x), X1), {same as INITIAL SYNCHRONIC CONDITION (INIT SYN COND) above at T1}
T3: Appear(Bullet(z), X4), AL(Bullet(z), X4), {INIT SYN COND}
T4: AL(Bullet(z), X5), {INIT SYN COND}
T5: AL(Bullet(z), X6), {INIT SYN COND}
T6: AL(Bullet(z), X7), {INIT SYN COND}
T7: AL(Bullet(z), X8), {INIT SYN COND}
T8: AL(Bullet(z), X9), {INIT SYN COND}
T9: AL(Bullet(z), X10), {INIT SYN COND}
T10: AL(Bullet(z), X11), Contact(Bullet(z), Object(z)), {INIT SYN COND}
T11: Damage(Object(z), X12), Disappear(Bullet(z)), {INIT SYN COND}

Figure 4: A SPECIFIC GSDO (Gun-Shooting-and-Damaging-Object) script, learned and captured from the event instance of Figure 3(a). The arrow indicates the value of RA (=0) that corresponds to Gun(x) pointing at Object(y).

For brevity, we have omitted some other visual parameters associated with Bullet, such as the RD (relative distance) between it and the Gun and Object, which would be changing as Bullet moves. If these values are included, the system might discover other regularities, such as whenever the Contact event happens, the RD between Bullet and target Object(y) would be 0.

Note that as indicated with an arrow in Figure 4, the Barrel is pointing straight at target Object(y), therefore the RA between the long axis of the Barrel and a line joining the centroid of Gun(x) to target Object(y) is 0.

Note also that underlying the time-step-by-time-step sequential structure of the script shown in Figure 4, there could be encodings of individual causal rules that were learned earlier or are learned as this current sequence of actions is observed or experienced. Based on the learning mechanisms of diachronic and synchronic causal conditions as described in Sections 2.2 and 2.3, some of the learned individual causal rules are:

Press-Trigger(Gun(x), X1)∧{INIT SYN COND}
➔ Appear(Bullet(z), X4)∧{INIT SYN COND} (9)

AL(Bullet(z),*X)∧{INIT SYN COND}
➔ AL(Bullet(z),*X+Δ1)∧{INIT SYN COND} (10)

AL(Bullet(z), X11)⋀Contact(Bullet(z), Object(z))
⋀{INIT SYN COND}
➔ Damage(Object(z), X12) ⋀Disappear(Bullet(z)),
⋀{INIT SYN COND} (11)

Eqn. 9 or causal rule 9 is shown to be a specific rule in the sense that the Press-Trigger and Appear actions both take place at specific locations X1 and X4 respectively. Eqn. 10 is general (i.e., the precondition is Bullet(z) at "any" X, represented as "*X" and the effect is such that Bullet(z) is at "the *same* any X plus one" location, represented as "*X+Δ1") because there is more than one instance of the bullet movement that have been observed and generalized over. Eqn. 11 is also specific. Eqns. 9 and 11 can become general rules after more instances of the actions involved are observed, such as after the situation of Figure 3(b) as represented in Figure 5.

Figure 4 is a SPECIFIC GSDO script because it contains specific values of various variables. Thus, it captures this particular event in which Gun(x) is shooting from a certain location, in a certain direction, and at target Object(y) at a certain location.

Now, suppose there is another instance of the event, shown in Figure 3(b), in which Gun(x) and target Object(y) are at different locations from that in Figure 3(a), then the script captured and learned will be as shown in Figure 5. In this instance, not only the locations of the objects involved are different, the distances between Gun(x) and target Object(y) are also different, resulting in Bullet(z) traveling for a shorter distance (fewer incremental distance steps) before hitting target Object(y).

T31: AL(Gun(x), X31), AL(Object(y), X40), AA(Barrel(Gun(x)), A130),
RD(Gun(x), Object(y), D6), RA(Barrel(Gun(x)), Object(y), 0)
T32: Press-Trigger(Gun(x), X31), {same as INITIAL SYNCHRONIC CONDITION
(INIT SYN COND) above at T31}
T33: Appear(Bullet(z), X34), AL(Bullet(z), X34), {INIT SYN COND}
T34: AL(Bullet(z), X35), {INIT SYN COND}
T35: AL(Bullet(z), X36), {INIT SYN COND}
T36: AL(Bullet(z), X37), {INIT SYN COND}
T37: AL(Bullet(z), X38), {INIT SYN COND}
T38: AL(Bullet(z), X39), Contact(Bullet(z), Object(z)), {INIT SYN COND}
T39: Damage(Object(z), X40), Disappear(Bullet(z)), {INIT SYN COND}

Figure 5: A SPECIFIC GSDO (Gun-Shooting-and-Damaging-Object) script, learned from the event instance of Figure 3(b).

After observing both instances of the GSDO event (each of which is learned as a SPEFCIFIC GSDO script), the system executes a "dual instance generalization" process as described above in Section 2.3, in which parameters that are observed to have different values are generalized, marked with a "*" which means "any value" in Figure 6 (such as in Eqn. 10). Figure 6 is thus the GENERAL GSDO script. It includes the knowledge of Eqn. 1 but is richer with more information. Therefore, "*X" for the AL of the Gun means "any value for X." The Press-Trigger predicate also has an argument *X, and this means it is an "any value" but the *same* value as the "X" in the AL predicate. Because in both instances, Bullet(z) appears at the mouth of the Barrel which is 3 locations away from its centroid, the generalization process takes note of this and generates *X+Δ3 for the location of the appearance of Bullet(z). Δ3 means Δ=3, which in turns means the location is incremented by 3 units from X, and it always has this value because it has this value in both instances of Figures 3(a) and 3(b).

*T: AL(Gun(x), *X), AL(Object(y), *XX), AA(Barrel(Gun(x)), *A),
RD(Gun(x), Object(y), *D), RA(Barrel(Gun(x)), Object(y), 0)
*T+M*Δ1: Press-Trigger(Gun(x), *X), {same as INITIAL SYNCHRONIC CONDITION
(INIT SYN COND) above at *T, 1st M=1}
*T+M*Δ1: Appear(Bullet(z), *X+Δ3), AL(Bullet(z), *X+Δ3), {INIT SYN COND}
*T+M*Δ1: AL(Bullet(z), *X+Δ3+N*Δ1), {INIT SYN COND} [**REPEAT** *N-TIMES, 1st N=1]
*T+M*Δ1: AL(Bullet(z), *X+Δ3+N*Δ1), Contact(Bullet(z), Object(z)), {INIT SYN COND}
*T+M*Δ1: Damage(Object(y), *XX), Disappear(Bullet(z)), {INIT SYN COND}

Figure 6: The GENERAL GSDO script derived from Figures 4 and 5.

Note that RA is 0 in both instances (indicated with an arrow in Figures 4 and 5), therefore it remains a "must have" value here, which is 0 (also indicated with an arrow in Figure 6). This means in a general situation Gun(x) must be pointing straight at target Object(y) before there is Damage to Object(y) at the end of the event.

There is one other kind of generalization process operating on other than the parameter values. This process operates on the *number of steps* of certain action, in this case, the movement of Bullet(z). Because in the two instances observed, the numbers of steps of location changes of Bullet(z) are different, the system creates a [REPEAT *N-TIMES, 1st N=1] instruction, which is to instruct that the particular predicate/action is to be executed "any" number of times, each time the value of N is to be incremented by 1, and the value of N starts at 1. This will generate the movement steps with their corresponding specific location values for any instantiated instance. The variable M for T has the same function as N – each time it is to be incremented by 1 and it begins with the value of 1. Because the temporal steps are incremented by 1 each time, the elemental incremental step is Δ=1 (i.e., represented by Δ1). If a variable is incremented by other than the amount of 1 each time, the learning process will encode it accordingly: Δ2, Δ3, etc. The learning process could conceivably also detect constant accelerated changes, or accelerated accelerated changes, if there are regularities as such. If the changes are random, then the system would encode the statistics such as the mean and deviation of the changes.

2.5. Script Instantiation for Prediction and Problem Solving

The instantiation of this GENERAL GSDO script of Figure 6, when a particular new event instance is encountered, would begin with setting the AL of Gun(x) and Object(y) to be at specific locations – i.e., *X and *XX will get these values – and the other values are then calculated accordingly. An instantiated version of the script can thus be used to predict: (a) that a Bullet(z) will appear; (b) the time at which the target Object(y) will be hit; (c) that Damage to Object(y) will result. The script can also be used in reverse for problem solving – if Damage to Object(y) is desired for Gun(x) and Object(y) at certain locations, this script could be selected as it encodes in its "outcome" - its last step - a Damage(Object(y), *XX) predicate. On invoking the script, it dictates that the Barrel of Gun(x) must point at Object(y) (RA…=0) in its first step, followed by a Press-Trigger action. Thus, this is returned as the solution to the problem.

3. Issues Related to General Causal Learning Framework

There is a number of issues related to the above causal learning mechanisms involving diachronic and synchronic causal conditions leading to the learning of event scripts and these will be discussed in this section.

3.1. The Availability of Basic Visual Information

The above process of learning diachronic and synchronic causal conditions requires the availability of various visual and physical parameters such as the absolute locations of various entities, the relative distances between entities, the absolute and relative orientations of the entities, etc. (Though not being used in the examples considered above, the constructs and structural dimensions of the entities involved would also be useful for formulating causal rules and event scripts.) It is reasonable to assume that this visual and physical information is available through the visual system. In fact, it has been argued and demonstrated in Ho [8] that without certain visual information, an agent would only be able to search randomly in the environment for food or other things to satisfy its needs, and its ability to survive would largely be dependent on chance. As a result, no "intelligent behavior" on their part could be emitted. Visual or other sensory information such as that above is essential for supporting the formulation of useful causal rules for intelligent behavior and survival. This is the indispensable essential causal role of visual and other sensory information. Thus, they have to be made available in an intelligent system to support intelligent behavior. The effort in naturally intelligent systems in evolving various sensory systems and the effort in including various sensory systems especially computer vision systems in AI systems attest to the importance of providing sensory information and accordingly, this information is available for the learning of causal knowledge.

3.2. The Identification of Event Script Boundary

The identification of event boundary (start and end states) is currently an open question [14, 15]. The sequence of activities depicted in Figures 3-6 could be part of a longer sequence of activities and we have not discussed how they may be "carved out" or isolated from the longer sequence and encoded as belonging to a particular script with the starting and ending boundaries as shown in the figures, even though the mechanisms of the learning of diachronic and synchronic causal conditions as described above could proceed independently of this issue.

In general, there are two competing requirements to forming and encoding a script from a sequence of actions. From the point of view of problem solving, if there is a long chunk of script that can immediately be retrieved to fit the current requirement (of a problem stated in the form of a start and goal state), the problem solving process will be expedient. However, shorter sequences of actions are useful in the sense that they are more transferrable between problem solving situations. For example, an entire Restaurant Script such as that investigated by Schank [4] consisting of quite a long sequence of actions can be activated expediently to deal with the problem of alleviating hunger by eating at a restaurant, but the shorter sequences inside the script, such as the Payment Sub-script, can be used in other problem situations if it is available separately. This issue merits further studies but below we describe and elaborate further the methods discussed in Ho [8] and propose some possible methods for this purpose:

1. If a sequence of actions is a result of problem solving, i.e., definite start and goal conditions were given earlier to a problem solving system and the system generates a sequence of actions as the solution, then that sequence of actions and the starting and ending states will be encoded in the script accordingly. This is a definitive way of bounding an event script as the start and end states are known and there is a purpose to the script as it serves a certain problem solving process.
2. If there is no activity in the environment for a relatively lengthy period of time before and after a sequence of actions, that sequence of actions is grouped into a script. (The problem with this criterion is the definition of "lengthy.")
3. Consider the example of the actions in Figure 4 that could be part of a longer sequence of actions. One

could begin with any change of state, such as the Damage of Object(y) and the Disappearance of Bullet(z) as one end point of a script (the "goal") and trace "backward" until the causes of the entity causing the Damage, namely Bullet(z), are all accounted for. Since Press-Trigger causes Bullet(z) to Appear, that, and perhaps the step before that that specifies the synchronic condition, could together be identified as the starting condition of the script. Searching further backward, one could conceivably find another "Acquire-Gun" script.

In conclusion, more future work is required to address the issues of script boundary.

3.3. Opportunistic Learning of Causality and Event Stream Separation

It has been shown in Ho and Liausvia [3] that in a "busy" environment in which many events are happening, the learning of causality is difficult. If an intelligent system fresh to the world is placed in a situation in which many never-before encountered events are happening, it would be difficult to learn what causes what. This is akin to what the psychologist William James said about an infant facing a "blooming and buzzing" confusion when he/she first experiences the world [16]. While it is not totally impossible to learn causality in a somewhat noisy environment, as shown by the work of Fire and Zhu [1] and Ho and Liausvia [2, 3], an intelligent system may actually have the opportunities to learn from a relatively quiet environment and then apply earlier learned knowledge to tease out the various streams of causalities in a "busy" environment. The situation is shown in Figure 7.

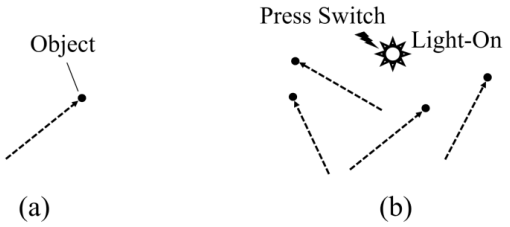

Figure 7: (a) Opportunistic situation: the movement of an object from a starting location to an ending location, learned and encoded as a Movement script. (b) Event stream separation in a busy environment: the movement of many objects simultaneously with a Press-Switch → Light-On event.

Figure 7(a) shows an ideal situation in which something like a Movement script could be learned – an object moving from a starting location to an ending location. This would be like the bullet movement segment of the script in Figure 4. If nothing else is happening like in Figure 7(a), then the time-step-by-time-step causality of the change of the absolute location of the object (like the AL's in Figure 4) could be learned and chained together into a Movement script through a process as described in Section 2. We are also assuming that there could be two similar instances of Figure 7(a) that take place at different locations and times so that a *general* Movement script is learned, like in the case of Figure 6 for the GSDO script.

However, had the system encountered the situation in Figure 7(b) first, in which there are many objects moving, and in addition there is a Press-Switch → Light-On event, it would be a situation of "blooming and buzzing" confusion – the movement of one object at one time instance could potentially be the cause of the movement of another object at the next time instance, etc. In the interest of not postulating too many spurious causalities, the system could be inhibited from attempting to learn causal rules when there are too many objects and actions involved, and there are no existing causal rules to discern the causalities between some of them.

However, in the event that the system had already learned a Movement script such as that in Figure 7(a), it could use that knowledge to achieve two things in the situation of Figure 7(b). Firstly, it can use the script to identify and separate the event streams in Figure 7(b) – there will be 4 streams of movement actions that can be matched to the Movement script. Here we are using an "explaining away" mechanism – if a certain action of a certain object could be causing a certain other action, as encapsulated in a known script, then that action's effect is accounted for and the system will not seek another action elsewhere to be causally linked to it. Likewise, if a certain effect could be accounted for by a known cause encapsulated in a certain script, then the system will not seek other possible causes elsewhere. This way the action/event stream identification and separation of the 4 movement tracks in Figure 7(b) could be achieved. Secondly, once the movement actions are accounted for by the 4 Movement scripts, the situation is no longer "blooming and buzzing" with many unknown causalities, and the Press-Switch → Light-On causality can then be learned. We believe this bootstrapping process is how the myriads of causalities in our seemingly complex environment are learned by natural intelligent systems and AI systems can also use the same mechanisms to learn these causalities.

3.4. Extraction of Functionality from Scripts

The GENERAL GSDO script of Figure 6 also encodes the operation and functionality of a gun. Earlier we conceived of Figure 6 as an event script. From the point of view of operation and functionality, we ask questions centered around the artifact, the gun. Let us consider the 5

"wh" questions that can be asked about the gun based on the GENEREL GSDO script (we are assuming that there is an extraction process that extracts the knowledge encoded in the script and provides the answers in English below):

1. *What* does a gun do? *Answer:* It can be used to damage an object across space/from a distance.
2. *How* does a gun work? *Answer:* You press the trigger and that will cause a bullet to be emitted from the barrel. The bullet will travel across space. Then the bullet, on contacting an object, will damage it. You have to point the gun at the object in order for the object to be damaged by the bullet.
3. *Where* can you operate a gun? *Answer:* Anywhere.
4. *When* can you operate a gun? *Answer:* Anytime.
5. *Who* can operate a gun? *Answer:* anyone with a hand that has fingers – this is not currently encoded explicitly in Figure 6 but presumably the Press-Trigger action entails this and this knowledge is encoded in other related scripts.

Thus, the GENERAL GSDO script of Figure 6 is not only an event script, but also encodes operational and functional information.

4. Summary and Discussion

We have described in this paper a principled way to treat causal conditions which is to separate them into two different types of conditions – synchronic and diachronic causal conditions. We also describe a principled framework of causal knowledge learning that enables long sequences of cause-effects to be learned and encoded into scripts for prediction and problem solving. A number of related issues are discussed which includes the kind of basic visual information that must be available for general learning of causal knowledge, the identification of event boundary to assist in the learning and encoding of scripts, opportunistic learning of causality and event stream separation, and the extraction of functionality from scripts. These issues should be further pursued in future work.

In a general situation, the structure of events in the environment can be learned and organized in the form of AND-OR graphs [9, 17-19]. The framework that we propose here for the learning and encoding of causal knowledge in the form of scripts can be combined with the methods of AND-OR graph learning [9, 17-19] to create a more robust and general causal learning framework.

Acknowledgement: This research is supported by the Social Technologies+ Programme funded by A*STAR Joint Council Office.

References

[1] A. Fire, and S.-C. Zhu. Learning perceptual causality from video. ACM Transactions on Intelligent Systems and Technology, 7(2):23, 2015. doi:10.1145/2809782.

[2] S.-B. Ho and Liausvia, F. (2016). A ground level causal learning algorithm. In Proceedings of the IEEE Symposium Series on Computational Intelligence for Human-like Intelligence, pp. 110-117, 2016.

[3] S.-B. Ho and Liausvia, F. (2016). On inductive learning of causal knowledge for problem solving. In Technical Reports of the Workshops of the 31st AAAI Conference on Artificial Intelligence, 2017.

[4] R. Schank and R. Abelson. Scripts, Plans, Goals and Understanding. Hillsdale: Lawrence Erlbaum Associates, 1977.

[5] N. Chambers and D. Jurafsky. Unsupervised learning of narrative event chains. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, pp. 789-797, 2008.

[6] M, Regneri, A. Koller, and M. Pinkal. Learning script knowledge with Web experiments. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 979-988, 2010.

[7] S.-B. Ho and F. Liausvia. Rapid learning and problem solving. In Proceedings of the IEEE Symposium on Computational Intelligence for Human-like Intelligence, pp. 110-117, 2014.

[8] S.-B. Ho. Principles of Noology: Toward a Theory and Science of Intelligence. Switzerland: Springer International, 2016.

[9] K. Tu, M. Meng, M. W. Lee, T. E. Choe, and S.-C. Zhu. Joint video and text parsing for understanding events and answering queries. IEEE Multimedia, 21(2):42-70, 2014.

[10] S.-B. Ho. On effective causal learning. In Proceedings of the 7th International Conference on Artificial General Intelligence, pp. 43-52, 2014.

[11] R. P. Abelson. The structure of belief systems. In R. C. Schank, and K. M. Colby (eds.) Computer Models of Thought and Language. San Francisco: W. H. Freeman, 1973.

[12] H. Jenkins and W. Ward. Judgment of contingency between responses and outcomes. Psychological Monographs, 7:1-17, 1965.

[13] R. A. Rescorla. Probability of shock in the presence and absence of CS in fear conditioning. Journal of Comparative and Physiological Psychology, 66: 1-5, 1968.

[14] T. F. Shipley and J. M. Zacks. Understanding Events: From Perception to Action. Oxford: Oxford University Press, 2008

[15] H. W. Leong and K. Kwok. Towards robust agent behaviors in modeling and simulation: Situation filling in with commonsense knowledge. In 20th Conference on Behavior Representation in Modeling & Simulation, pp. 41-48, 2011.

[16] W. James. The Principles of Psychology. New York: Dover, 1950.

[17] A. Gupta, P. Srinivasan, J. Shi, and L. S. Davis. Understanding videos, constructing plots: Learning a visually grounded storyline model from annotated videos. In CVPR 2009.

[18] Z. Si, M. Pei, B. Yao, and S.-C. Zhu. Unsupervised learning of event AND-OR grammar and semantics from video. In ICCV 2011.

[19] M. Pei, Y. Jia, and S.-C. Zhu. Parsing video events with goal inference and intent prediction. In ICCV 2011.