Christopher Thomas

Exercise 4: The purpose of this exercise is to apply multimedia functional dependency to multimedia applications design. Given an application (its requirements), design the multimedia database using multimedia functional dependency theory. Then specify the patterns (IC cards) associated with the multimedia database. The application is the personal health care system that allows the user (a senior citizen) to access related multimedia documents using gestures. A new classification scheme based upon the gestures associated with the multimedia documents is to be introduced. This would allow users to search for multimedia documents similar to a known audio search key (such as the voice of a certain author). Your task is to design the multimedia database and associate patterns (IC cards), which can in turn be transformed into IC index and finally an implementation.

Before we can embark on designing the multimedia database, we must first develop a mapping relation (functional dependency relationship) between the elements that we will be dealing with in the database. We will assume here that the gesture is provided by an interface such as Kinnect, which can provide us with multiple types of data. Thus, the "gesture" is a composite, consisting of some unique gesture identifier (perhaps a GUID), "sound data" (if any), user's motion data (from their movements in front of the Kinnect), facial expression (such as mouth open / closed), and a gesture descriptor. Note that the primary key for each entry in the gesture table is the unique gesture identifier, but this unique gesture identifier does not allow us to map gestures between documents; it only creates a unique primary key for the relation. The gesture descriptor provides the nexus between the components of the system (documents and gesture relation).

Thus, the gesture descriptor of each tuple is functionally dependent upon the sound data, motion data, and the facial expression of that tuple. Note that multiple sound data, motion data, and facial expressions may produce the same gesture descriptor (if they are sufficiently similar). Thus, the gesture descriptor can be seen as the homological mapping of the sound data, motion data, and facial expression data using some function *f* into an *n*-dimensional Hilbert space using a technique such as PCA, ICA, etc. More formally, the gesture descriptor has a type-M functional dependency on the sound, motion, and facial expression because it is the homological image of those attributes. Thus, we obtain the first multimedia functional dependency relationship:

$$\{Sound, Motion, Facial\}_{f_1(t_1)} \rightarrow \{Gesture\ Descriptor\}_{f_2(t_2)}$$

This MFD states that for any two tuples *a* and *b* such that a(sound,motion,facial) is considered similar within the threshold $t_1$to b(sound,motion,facial) by the function f_1 then a(gesture descriptor) is considered similar within the threshold t_2 to b(gesture descriptor) by the function f_2.

Note, before the query is done evaluating, we check another relation in the database, the Map relation. The map relation contains mappings from gesture descriptors to document GUIDs (globally unique identifiers). The map relation has a primary key of gesture descriptor (i.e. gesture descriptors uniquely determine the tuple). Thus, note that the gesture descriptor in the Gesture relation is actually a foreign key of the Map relation. Thus, before the gesture can be inserted, the gesture descriptor generated from the gesture relation is first checked against the map relation to determine if it is already there. If not, a new tuple is inserted with the gesture descriptor (so as not to break the foreign key constraint of the Gesture relation). The document

GUID(s) associated with the newly inserted gesture descriptor now need to be determined using the Document relation. Note that the relationship between the gesture descriptor in the Gesture relation and the gesture descriptor in the Map relation are the full strength primary / foreign key relationship; as we are dealing with equality, we do not need the relaxed multimedia functional dependency relationship between them and instead require traditional primary / foreign key equality.

The Document relation will be used to determine the documents that the gesture descriptor in the map relation maps to. The Document relation consists of tuples with attributes of document type, document guid, document content, and a document descriptor. Note that the document GUID uniquely identifies each tuple in the relation, and thus the primary key of the relation is the document GUID. Thus, for each gesture descriptor, the Map relation contains 1 or more document GUIDs (which is a foreign key from the document relation).The document descriptor works analogously to the gesture descriptor described above, except that the document descriptor is dependent on the document content rather than on gesture content. Thus, we arrive at our second multimedia functional dependency:

$$\{Document\ Content\}_{f_3(t_3)} \to \{Document\ Descriptor\}_{f_4(t_4)}$$

This MFD states that for any two tuples *a* and *b* such that a(document content) is considered similar within the threshold $t_3$to b(document content) by the function f_3 then a(document descriptor) is considered similar within the threshold t_4 to b(document descriptor) by the function f_4.

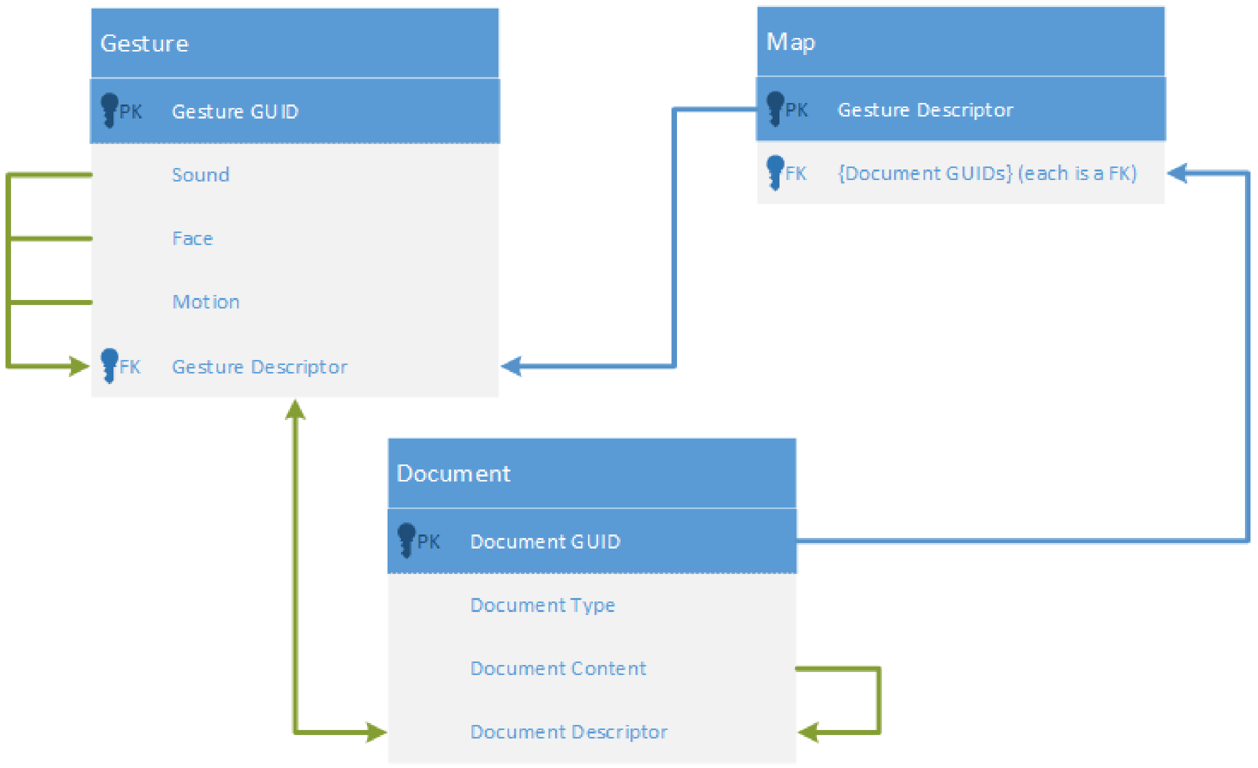
While this may seem complete, it actually isn't. We are still missing the mapping from the gesture descriptor to the document descriptor, i.e. when a new gesture descriptor comes in, how can we associate it with documents which it should match with? Thus, we have an implicit multimedia functional dependency between these two values. We note that both the document descriptor and the gesture descriptor are both vectors in an *n*-dimensional Hilbert space. Thus, similar gesture descriptors (within some threshold) will map to the same document(s) (based on another threshold). Note that we assume for simplicity here that a simple vector distance function – the Euclidean Distance – exists for computing the similarity of vectors, but that the thresholds will be different for each. Thus, we arrive at the final multimedia functional dependency in our database:

$$\{Gesture\ Descriptor\}_{EUCLIDEAN(t_5)} \leftrightarrow \{Document\ Descriptor\}_{EUCLIDEAN_{(t_6)}}$$

Note that this MFD is transitive. We can now see that when a new gesture is inserted, an entry in the Map is created (if one didn't already exist) using this relation between gesture descriptor and document descriptor to determine which document guids to list in the Map. Note also that when a new document is inserted, a lookup process can occur based on gesture descriptors and insert that document into the list of document GUIDs associated with that gesture descriptor.

(see next page)

From the above discussion, we arrive at the following *database schema* describing the multimedia database:



In the illustration, green lines indicate a multimedia functional dependency based on the previous functions given. Blue lines illustrate other typical database relationships, such as primary key, foreign key, etc.

Given the database schema above, we can now give a system description using IC Cards. The IC cards will examine the steps at a high level of detail, walking us through all the steps necessary to recognize a gesture and use the system as defined here.

IC Card IC Name: Process Gesture

Description: Interpret the gesture, query the database, and return documents to the user
Interaction Pattern:

Mixed

Time Critical Condition: N/A
Name of Other IC: Sub-Components
Message to Other IC: None
Other IC's Task: None
Card 1 of 1 (If necessary please use several IC cards to describe an IC)

This card represents the top of the tree of processing in the database (it represents the entire operation). We now will get into the details of the sub-components.

IC Card IC Name: Gesture Capturer

Description: Capture the gesture (sounds, face, movement) and calculate the gesture descriptor
Interaction Pattern:

By Myself with Interaction

Time Critical Condition: As soon as possible after gesture detected
Name of Other IC: Fetcher
Message to Other IC: Fetch documents for gesture descriptor X
Other IC's Task: Check the map for the gesture descriptor and return documents. If not there, compute
Card 1 of 1 (If necessary please use several IC cards to describe an IC)

IC Card IC Name: Fetcher

Description: Check Map for the descriptor X and GUIDs. If descriptor not there, contact comparitor and check map again. Return document content
Interaction Pattern:

By Myself with Interaction

Time Critical Condition: Before returning documents to user
Name of Other IC: Comparitor
Message to Other IC: Descriptor X not in Map, insert it with document GUIDs
Other IC's Task: Find document content relevant to the gesture descriptor and insert it into the map
Card 1 of 1 (If necessary please use several IC cards to describe an IC)

IC Card IC Name: Comparitor

Description: Calculate comparison functions on gesture descriptor and document descriptors to find matches and insert into Map relation
Interaction Pattern:

By Myself no Interaction

Time Critical Condition: Before Fetcher rechecks the Map relation
Name of Other IC: None
Message to Other IC: None
Other IC's Task: None
Card 1 of 1 (If necessary please use several IC cards to describe an IC)

IC Card

IC Name: Inserter

Description: Given a new document, compute the document descriptor and insert it into the document relation and insert it into all matching gesture descriptor maps in the Map relation
Interaction Pattern:

By Myself no Interaction

Time Critical Condition: As soon as the document arrives
Name of Other IC: none
Message to Other IC: none
Other IC's Task: none
Card 1 of 1 (If necessary please use several IC cards to describe an IC)

Note: The inserter is used only when documents are inserted into the relation.

IC Card

IC Name: Retriever

Description: Performed User Side - Given a document GUID(s) query the document relation and retrieve the document content and display it
Interaction Pattern:

By Myself no Interaction

Time Critical Condition: After user receives document GUID(s) back
Name of Other IC: None
Message to Other IC: None
Other IC's Task: None
Card 1 of 1 (If necessary please use several IC cards to describe an IC)