온톨로지를 이용한 텍스트 애니메이션 객체 탐색

Searching Animation Models with a Lexical Ontology for Text Animation

장은영, Eunyoung Chang, 이희진, Hee-Jin Lee, 박종철, Jong C. Park
한국과학기술원 전자전산학과 전산학 전공

요약 텍스트 애니메이션 시스템에서는 자연언어 단어로 표현된 개체들을 한정된 수의 애니메이션 모델로 나타낸다. 그러나 자연언어 단어의 수에 비해 기존의 모델 DB 에 있는 모델의 수가 훨씬 적은 것이 일반적이기 때문에 해당 단어에 대응되는 애니메이션 모델이 존재하지 않는 경우가 있게 된다. 이러한 경우, 해당 단어가 가지는 의미를 최대한 보존할 수 있는 대체 모델을 찾을 수 있는 방법이 필요하다. 본 논문은 애니메이션에서 캐릭터 또는 사물로 표현되어야 하는 명사에 대해, 온톨로지에서 해당 명사와 상위(hypernym), 하위(hyponym), 부분(member meronymy) 관계에 있는 다른 단어를 탐색하여 적절한 모델을 찾는 방안을 제안한다.

핵심어: Text Animation, Natural Language Processing, Ontology

1. 서론

텍스트 애니메이션은 이야기 글의 내용을 분석하여 3D 애니메이션으로 변환하는 연구로서[1], 누구나 자연 언어를 통해 쉽게 애니메이션을 생성할 수 있게 하는 것을 목적으로 한다. 관련된 연구로는 자연 언어 문장 묘사에 대한 장면을 생성하는 연구를 비롯하여[2] 교통 사고 보고서로부터 재연 애니메이션을 생성하는 연구가 있으며[3], 최근 동화나 영화 대본의 간단한 문장으로부터 애니메이션을 생성하여 스토리텔링에 활용하려는 연구가 이루어지고 있다[1,4,5].

이러한 연구들은 아직 초기 단계로서, 애니메이션에서 보여질 캐릭터와 동작을 미리 정해진 상세도에 맞게 묘사해야 하는 등 입력으로 고려하는 자연 언어 문장의 범위가 매우 한정적이고, 임의의 자연 언어 어휘에 대응하는 적절한 애니메이션 데이터를 찾는 방법에 대해서는 충분히 고려하지 않고 있다. 따라서 보다 일반적인 단어 및 자유로운 수준의 묘사에 대해 의미를 제대로 전달하면서 애니메이션으로 보여줄 수 있는 방법의 연구가 필요하다.

본 논문은 애니메이션에서 캐릭터 또는 사물로 표현되어야 하는 명사에 대하여, 온톨로지의 계층 정보를 활용하여 애니메이션에 적합한 대체 단어를 찾는 방법을 제안한다. 온톨로지는 여러 개념 간의 관계 정보를 가지고 있으므로 이를 이용하여 최대한 유사한 의미의 대체 단어를 탐색할 수 있다. 본 논문에서는 특히 어휘 온톨로지(lexical ontology)를 사용하여 얻을 수 있는 결과를 보이고, 애니메이션 생성의 관점에서 이러한 어휘 수준의 정보를 이용하는 방법의 가능성과 그 한계에 대해 논의한다.

본 논문의 구성은 다음과 같다. 2절에서 텍스트 애니메이션 관련 연구들을 소개하고, 3절에서 본 논문에서 다루고자 하는 문제를 정의하고, 접근 방식을 설명한다. 4절에서는 알고리즘을 제안하고, 5절에서 구현된 결과와 관련된 논의를 하고, 6절의 결론과 향후 계획으로 마친다.

2. 관련 연구

텍스트 애니메이션 관련 연구들은 기존에는 주로 자연언어로 기술된 문서 내용에 대한 시뮬레이션 목적의 연구들이 진행되었으며[3], 스토리텔링을 목적으로 자연 언어 문장 또는 문장의 시퀀스를 입력으로 받아 애니메이션을 생성하는 연구도 이루어지고 있다[1,4,5].

본 연구의 선행연구에서는 텍스트 형태의 동화로부터 멀티미디어 동화를 자동으로 생성하기 위해 고려해야 하는 요소들을 크게 캐릭터 제어와 카메라 제어 등으로 나누어 분석하고, 시간 관리, 참조 현상 해결, 캐릭터의 세부 동작 결정, 카메라 워크 및 장면 전환 기법에 관한 연구를 진행하여 왔다. 특히 참조 현상 해결을 위해 WordNet의 일부 데이터를 차용하여 유의어와 상, 하위어를 검색하는 데에 사용하기도 하였는데, 이를 애니메이션

데이터를 찾는 과정까지 적극적으로 활용하지는 않았다[1]. 본 논문은 이 연구를 확장하려는 것으로, 기존 시스템에서 너무 긴밀하게 연결되어 있는 '자연 언어 단어'-'의미'-'모델'의 관계를 다소 느슨하게 하여, 입력으로 받을 수 있는 텍스트의 한계를 극복하고자 한다.

ScriptViz는 학생들의 작문 결과를 시각화해주는 것을 목적으로 하는 시스템으로, 실시간 입력으로 들어오는 자연 언어 문장의 의미를 분석하고, 참조 현상 해결, 동작의 선행 조건 체크, 위치 설정을 차례로 거쳐 애니메이션으로 시각화한다[4]. 시스템에서 목적에 맞게 제작한 일종의 상식 사전을 이용하고 있으나, 일반적인 입력을 처리하기에는 사전이 매우 제한적인 것으로 보인다.

Confucius는 스토리텔링을 목적으로 하여, 앞 단계의 시스템에서 자동으로 이야기를 생성하는 것을 가정하고, 이와 같이 생성된 이야기를 애니메이션으로 보여주고자 하는 시스템이다[5]. 사람의 동작을 묘사하는 문장을 대상으로, visual role, somatotopic effectors, level-of-detail에 따라 동사를 분류하여, 각각에 따라 논항을 적절히 보여주려 하고 있다. 그러나 다양한 데이터를 분석하는 대신 개념적으로만 문제에 접근하고 있을 뿐만 아니라, 본 논문에서 지적한 것과 같은 문제는 다루지 않고 있다.

위에서 살펴본 연구들은 모두 기초 단계로서 자연 언어 문장으로부터 애니메이션을 생성하기 위해 파악해야 할 정보를 정의하고, 시스템으로의 구현 가능성을 보였다는 점에서 의의가 있다. 반면에 충분히 많은 데이터를 고려하지 못하여, 비현실적인 가정으로 문제를 단순화시키는 경향을 보이기도 한다. 특히 명사나 동사 등의 자연 언어 어휘에 일치하는 애니메이션 데이터가 이미 존재함을 가정하고 있는데, 이는 일상적인 자연 언어를 입력으로 받는 시스템에서는 현실적으로 불가능하며, 외부 자원을 충분히 활용하여 다양한 자연 언어 어휘에 대해 적합한 데이터를 찾는 방법을 모색해야 한다. 본 연구에서는 기존에 널리 쓰이고 있는 온톨로지 자원을 사용하여, 자연 언어 어휘로 표현할 수 있는 개념과 애니메이션으로 보여줄 수 있는 개념의 차이를 파악하고 이를 효과적으로 반영할 수 있는 방법을 제안한다.

3. 문제 정의

본 연구에서 다루고 있는 텍스트 애니메이션 시스템은 <그림 1>과 같이 구문분석기, 명령 생성기, 애니메이션 생성기의 세 부분으로 이루어져 있다. 전체 시스템에서 자연 언어가 처리되는 과정은 다음과 같다. 자연 언어 문장이 처음 입력되면 구문분석기가 논항 구조를 분석하여 애매성을 해소하고 생략된 정보를 채운 상태의 의미를 생성한다. 명령 생성기는 입력된 문장의 의미를 구성하는 각 의미역(semantic role)에 대해 불러올 애니메이션 데이터를 따로 결정한 뒤, 이들을 보여주는 방식(시간, 위치, 방향 등)을 고려하여 결과적으로 각 캐릭터와 사물의 구체적인 동작을 지정하는 명령 스크립트를 생성한다. 마지막으로 애니메이션 생성기는 스크립트를 읽고 애니메이션 DB를 참조하여 결과 애니메이션을 생성한다.

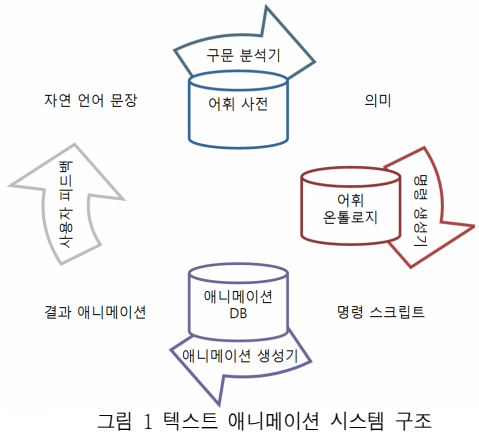


그림 1 텍스트 애니메이션 시스템 구조

명령 생성기에서 주어진 의미에 정확히 일치하는 애니메이션 데이터가 존재하지 않을 경우에 온톨로지를 탐색하여 대체 데이터를 찾아 이를 스크립트에 사용한다. 주로 캐릭터나 사물을 지칭하는 명사에 대해서는 주어진 단어의 의미에 대해 온톨로지를 탐색하여 대응되는 애니메이션 모델의 파일 이름을 찾게 된다. 사건의 발생을 기술하는 동사의 경우에는 논항과 합쳐져서 온전한 의미를 갖게 되므로, 동사뿐만 아니라 논항의 의미를 함께 고려하여 온톨로지를 탐색한 뒤, 구체적인 동작을 실행할 애니메이션 모델 및 동작의 이름을 찾아야 한다.

3.1절에서는 자연 언어 어휘와 일치하는 애니메이션 데이터가 존재하지 않는 경우를 분석하고, 3.2절에서는 대체 데이터를 찾기 위해 온톨로지를 탐색하는 과정에서 고려할 점에 대하여 논의한다.

3.1 자연 언어 어휘와 애니메이션 데이터

자연 언어 어휘와 일치하는 애니메이션 데이터를 찾을 수 없는 경우는 크게 아래의 두 가지가 있다.

(1) 텍스트 애니메이션 시스템에서는 자연언어 단어의 수에 비해 기존의 모델 DB에 있는 모델의 수가 훨씬 적기 때문에 해당 단어에 대응되는 애니메이션 모델이 존재하지 않는 경우가 있게 된다. 애니메이션 모델은 수작업으로 개발되고 있는데, 시스템에서 입력으로 받을 수 있는 자연 언어 어휘의 범위를 제한하지 않는다면 개발 초기 단계에서는 항상 문제가 발생하게 된다.

예를 들어 다양한 종의 새가 등장하는 애니메이션을 만들고 싶더라도, 새에 대해 특화된 애니메이션 DB를 사용하고 있는 것이 아니라면, 한 두 종류의 새에 대한 모델만을 사용할 수 밖에 없을 수 있다.

(2) 발화자가 동일한 주제의 이야기를 하고자 할 때, 자연 언어를 통해 서술하는 경우와 애니메이션을 통해 보

여주는 경우에 사용하는 개념들 간에 차이가 있을 수 있다. 자연 언어를 통해 서술하는 경우에는 집합적이거나 추상적인 개념을 많이 사용하고, 겉으로 보여지는 사건을 나열하기보다는 주로 인과 관계를 드러내기 위해 인물의 순간의 생각과 감정을 기술하는 편이다. 그러나 이러한 개념들을 애니메이션으로 보여줄 때는 각각 구체적이고 개별적인 개념으로, 즉 인물의 내적 상태를 겉으로 보여줄 수 있는 동작으로 보여주어야 한다.

예를 들어 'monster'에 대한 이야기가 진행된다면, 이를 실제로 보여줄 때는 'troll'이나 다른 괴물을 등장시켜서 'monster'라는 느낌을 전달할 수 있어야 하고, 집합체인 '가족'을 애니메이션으로 보여주려면 개별 가족 구성원을 따로 불러서 이를 한데 보여주어야 한다. 또한 인물이 화가 났다거나 고민이 있다고 할 때, 애니메이션에서는 이를 표정과 몸짓으로 보여주어야 한다.

위와 같은 경우에 적절한 대체 데이터를 찾기 위해서는 각 개념들 간의 관계에 대한 정보를 별도로 가지고 있어서 유사한 개념, 구체적 개념, 개별적 개념, 그리고 내적 상태와 관계된 외적 표현 등에 대한 정보를 찾을 수 있어야 한다. 이 중에서 마지막을 제외하고는 어휘 온톨로지를 탐색하는 것으로 가능하며, 마지막의 경우는 시각적인 표현에 대해 특화된 지식 사전을 따로 구축하고 이를 참조하는 방식으로 해결하여야 한다. 본 논문에서는 객체 탐색을 위주로, 어휘 온톨로지를 활용하는 측면에서 앞으로의 논의를 진행하고, 지식 사전을 구축하는 문제는 향후에 다루도록 한다.

3.2 온톨로지의 정보

본 논문에서는 온톨로지의 유의 관계(synonymy), 상위 관계(hypernymy), 하위 관계(hyponymy)의 정보를 주로 이용하고, 추가로 명사에 대해서는 부분 관계(member meronymy)를 이용한다.

캐릭터 또는 사물을 지칭하는 명사에 대해서는 추상적이거나 구체적인 개념으로 대체 데이터를 찾기 위해 하위어와 상위어를 탐색하고, 특히 집합적인 개념을 지칭하는 복수 개체 명사에 대해서는 부분어를 파악하여 개개의 모델를 따로 나타냄으로써 해당 명사의 의미를 나타낸다. 동사는 논항까지 고려한 단어의 조합을 가지고 대체 의미를 찾아야 한다. 그러나 대부분의 온톨로지는 동사의 여러가지 의미를 각 쓰임에 따라 구분하기는 하나, 이들을 적절히 구별하여 사용할 만한 정도로 충분한 정보는 제공하지 못하고 있다.

온톨로지를 탐색할 때, 기준이 되는 개념에서 상위로 이동하면 추상적인 개념, 하위로 이동하면 구체적인 개념이 된다. 이를 다르게 이해하면, 상위로 이동하면 원래 단어의 의미 중 일부만을 의미하는 단어를 찾게 되어, 정보를 일부 잃어버리는 결과가 된다. 반면 하위로 이동하면 원래 단어의 의미를 모두 포함하면서 다른 의미를 덧붙이는 것으로, 기존의 정보를 잃지는 않지만, 추가의 정보를 포함하게 될 가능성이 있다. 애니메이션에서는 보는 사람에게 시각적으로 풍부한 정보를 제공하는 것이 더 낫다고 판단되어, 본 연구에서는 되도록 구체적인 모델과 동작 데이터를 보여주기로 한다. 이와 같은 이유에서 본 연구에서는 하위어를 상위어보다 먼저 탐색하는 것을 우선으로 한다. 이러한 탐색을 여러 번 반복하게 되는 경우, 상위어와 하위어를 번갈아 가면서 탐색하게 되면 원래 의미의 일부를 잃고 없던 의미를 추가하게 되는데, 이러한 탐색이 반복될수록 원래 의미와 대립되는 의미를 찾을 위험성이 커진다. 따라서 본 연구에서는 이러한 탐색 방식은 되도록 지양하도록 하였다. 한편 유의어는 비교적 비슷한 수준의 정보를 담고 있어 의미의 전이에 따른 위험성을 적게 가지므로, 우선적으로 탐색되도록 하였다.

본 연구에서는 WordNet[6]을 대상 온톨로지로 사용한다. WordNet은 엄밀한 의미에서는 온톨로지로 분류되지 않고 있지만, 명사 간의 유의, 상위, 하위, 부분 관계를 표시하고 있으며, 비교적 넓은 범위의 단어들을 많이 포함하고 있어 본 연구에서 사용하기에 적합하다.

4. 온톨로지 탐색 알고리즘

본 절에서는 입력으로 주어진 단어에 일치하는 모델이 DB에 존재하지 않을 경우, 그 의미를 가장 잘 전달하는 대체 모델을 선택하기 위해 WordNet 탐색을 활용하는 알고리즘을 설명한다.

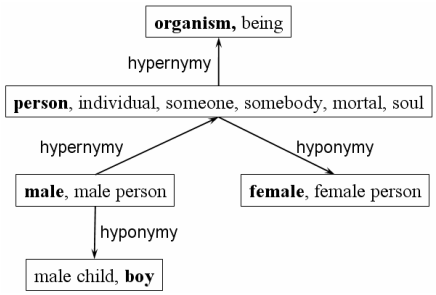


그림 2 온톨로지의 일부 – 단수 개체 지시어의 예

주어진 단어가 단수 개체 지시어인 경우, 온톨로지에서는 해당 단어의 상위어와 하위어를 찾을 수 있게 된다. 3.2절에서 설명한 바와 같이, 하위어는 원래 단어의 의미를 잃지는 않는 반면, 상위어는 원래 단어의 의미 중 일부만을 의미하므로 상위어보다는 하위어들을 먼저 재귀적으로 탐색하면서 적절한 애니메이션 모델을 검색한다. 예를 들어, male에 대한 애니메이션 모델이 존재하지 않을 때, <그림 2>의 온톨로지를 참고하여 male의 하위어인 boy의 모델을 우선적으로 선택하는 것이 상위어인 person에 대한 모델을 선택하는 것에 비해 성별 정보를 보존하고 있으므로 더 나은 결과를 가져온다.

해당 단어의 하위어들 중에서 적절한 모델을 선택하는 과정이 실패한 경우에는 상위어를 찾아 대응되는 애니메이션 모델을 검색한다. 이것도 실패할 경우에는 크게 두 가지 방법으로 접근할 수 있는데, 첫번째 방법은 상위어만을 재귀적으로 탐색하는 것으로, 계속적으로 의미가 일반화되어 결국 전달력이 떨어지게 되고, 두번째 방법은 상위어에 대해 다시 하위어부터 탐색해 나가는 것으로, 첫번째 방법에 비해 의미가 덜 일반화되지만 원래 단어의 의미에 배타적인 의미를 포함하는 단어를 선택하게 될 위험이 있다. 앞의 예제에서 만약 boy에 해당하는 애니메이션 모델이 없다면, <그림 2>의 온톨로지를 참고하여 상위어인 person의 모델을 찾는다. 그러나 이런 모델도 없을 경우, person의 상위어인 organism을 선택하면 전달력이 현저히 떨어지고, 하위어인 female을 선택하면 종종 모순이 있는 애니메이션을 생성하게 된다. 본 연구의 시스템에서는 모순이 없는 애니메이션을 생성하기 위해 첫번째 방법을 우선적으로 선택한다.

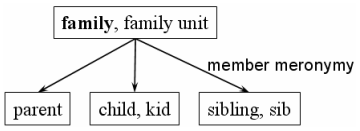


그림 3 온톨로지의 일부 – 복수 개체 지시어의 예

명사가 복수 개체 지시어인 경우에는 기본적으로 단수 개체를 나타내고 있는 애니메이션 모델들을 여러 개 선택하여 주어야 한다. 이는 해당 명사의 부분어들을 탐색하여 복수 개체를 단수 개체로 분리함으로써 가능한데, 각 부분어에 대해서는 본 절에서 설명하고 있는 알고리즘을 처음부터 다시 적용하여 애니메이션 모델을 선택해 주어야 한다. 예를 들어 family에 대한 모델을 찾으려고 할 때, <그림 3>의 온톨로지에서 family의 부분어인 parent, child, sibling에 대해서 다시 적절한 모델을 찾아서 보여주는 것이다. 그러나 이러한 부분어가 온톨로지에 존재하지 않는 경우에는, 원래 단어의 하위어의 부분어를 찾도록 한다. 이 과정도 실패한다면 원래 단어의 상위어를 찾아 위 과정을 반복한다.

전체 알고리즘에 대한 수도 코드는 <그림 4>와 같다. 앞에서 살펴본 것과 같이 일반적인 단어에 대해 애니메이션 모델DB에 적절한 모델이 없어 임의의 모델을 보여주어야 하는 경우에도, 온톨로지를 효과적으로 활용하면 의미를 최대한 보존하는 모델을 찾아서 보여줄 수 있다.

5. 결과 및 논의

앞 절에서의 온톨로지 탐색 알고리즘은 Python으로 구현되었으며, 온톨로지는 WordNet 2.0을 사용하였다. 애니메이션 DB는 애니메이션 모델 및 동작 파일들의 모음으로 이루어져 있는데, 각 모델 파일의 이름에 그 모델을 뜻하는 단어를 나타내어 검색에 활용하였다. 파일 이름을

```
word = user_input()
senses_of_word = word.getSenses()
sense = WSD(senses_of_word)
result = for_any_senses(queue([sense]))
return result

function for_any_senses(senses):
    if senses.empty():   return [ ]
    else:
        models = nil
        sense = senses.pop()
        if sense represents multiple objects:
            model = for_multi(sense)
        else:
            model = for_non_multi(sense)
        models = for_any_senses(senses)
        return append([model], models)

function for_multi(sense):
    model = search_for_model(sense)
    if model != nil:
        return model
    else:
        if sense.meronyms != nil:
            return for_any_senses(sense.meronyms)
        else:
            model = for_multi_hyponym(sense.hyponyms)
            if model != nil:
                return model
            else:
                return
for_multi_hypernym(sense.hypernyms)

function for_multi_hyponym(senses):
    if senses.empty(): return [ ]
    else:
        sense = senses.pop()
        model = search_model(sense)l
        if model != nil: return model
        else:
            if sense.meronyms != nil:
                return for_any_senses(sense.meronyms)
            else:
                senses.append(sense.hyponyms)
                return for_multi_hyponym(senses)

function for_multi_hypernym(senses):
    similar to ‘for_multi_hyponym’

function for_non_multi(sense):
    similar to ‘for_multi’
```

그림 4 온톨로지 탐색 알고리즘

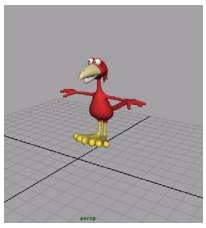
만드는 규칙은 다음 (1)의 정규식과 같다. 여기서 MODELNAME 은 해당 파일이 담고 있는 애니메이션 모델의 이름이고, DESCRIPTION은 해당 모델을 설명하는 명사이다. 해당 모델의 상, 하위 개념에 해당하는 명사들은 DESCRIPTION에 넣지 않도록 하였다. (2)는 본 시스템에서 사용하는 모델 DB의 파일명 예시이다.

(1) MODELNAME(_DESCRIPTION)*.mb

(2) dennis_boy.mb
frankie_troll.mb
mansuit_man_father.mb
redbird_bird_red.mb

현재 시스템에서 가지고 있는 모델 파일은 동물 및 인물에 대한 카툰 캐릭터 40개로, 개, 고양이, 새, 돼지, 소년 등 동화에 자주 등장하는 캐릭터로 구성되어 있다. 이를 가지고 3.1절에서 언급된 상황에 대해 실제로 탐색해 본 결과는 아래와 같다.

(상황 1) skylark 입력
→ skylark 검색 실패
→ (하위어 없음)
→ 상위어 bird 검색 성공
→ redbird_bird.mb



(상황 2) monster 입력
→ monster 검색 실패
→ 하위어 troll 검색 성공
→ frankie_troll.mb

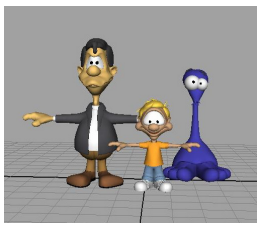
(상황 3) family 입력
→ family 검색 실패
→ 부분어 child, parent, sibling 각각 검색

child 검색 실패
→ 하위어 boy 검색 성공
→ dennis_boy.mb

parent 검색 실패
→ 하위어 father 검색 성공
→ mansuit_man_father.mb

sibling 검색 실패
→ 하위어 모두 검색 실패
→ 상위어 organism 검색 성공

→ dennis_boy.mb,
mansuit_man_father.mb,
pompis_organism.mb



위에서 살펴 본 것과 같이 입력된 단어에 정확히 일치하는 모델이 DB에 존재하지 않더라도 유사한 개념의 모델을 찾는 것이 가능하다. 문장의 각 명사에 대하여 이러한 과정을 거쳐 적절한 모델을 찾으면, 동사를 이용하여 검색된 동작 데이터와 합쳐져서 애니메이션을 만들게 된다.

그러나 (상황 3)에서 parent라는 부분어가 있을 때 이를 곧바로 father로 연결하여 mother를 보여줄 여지를 없애거나, sibling과 연관된 대부분의 개념들이 애니메이션 DB에서 발견되지 않아 매우 상위에 위치하는 organism을 보여주게 되어 직관적이지 않은 결과를 보여주는 등 어휘 온톨로지를 사용하는 것만으로는 모든 문제를 풀 수 없다는 것를 보여주기도 한다. 어휘 온톨로지를 사용하는 방법은 다음과 같은 한계를 가지고 있다.

(1) 집합 개념이 주어졌을 때, 각 부분어에 대한 모델들을 모두 보여주어야 하는지, 또는 한두 개의 부분어에 대한 모델을 여러 개 보여주어야 하는지의 문제는 해당 집합체의 성격에 따라 그 답이 달라진다. 만일 troops가 입력되었다면, soldier에 대한 하나의 모델을 여러 개 보여주는 것이 훨씬 효과적일 수 있다. 또한 family-parent와 같이 해당 집합체에 속하는 부분어의 개수가 일반적으로 정해져 있는 경우도 있다. 이와 같이 주어진 의미에 부합하는 모델을 찾는 것에 더하여 '상식적인 선에서 효과적으로' 표현하기 위해서는, 보다 상식 정보를 많이 포함하도록 변형된 온톨로지를 사용하여야 한다.

(2) 어휘 온톨로지의 구조가 잘 정의되어 있지 않으면, 유사한 개념을 찾을 가능성이 적어진다. 한 예로 WordNet에서 명사는 하나의 트리 구조로 이루어져 이에 속하는 모든 단어가 공통의 최상위어 entity를 가지는 데 반해, 동사는 매우 많은 트리 구조들로 이루어져 있고, 이들 구조는 또 여러 개의 파일들로 묶여 있다. 이러한 구조로 인해, say-talk와 같이 애니메이션 상으로 보여지는 동작은 유사하지만 최상위어는 각각 express-act로 달라 온톨로지 탐색만으로는 두 개념 간의 관계를 찾을 수 없는 경우도 생기게 된다. 따라서 온톨로지의 활용 목적에 따라 유의어의 범위를 확장하고 구조를 통합하는 등, 관계된 개념을 충분히 찾을 수 있도록 추가적인 작업이 필요하다.

(3) 명사 입력에 대해서는 제안한 방법이 가능성이 있음을 보였으나, 동사에 대해서는 다른 방식의 접근이 이루어져야 한다. 3절에서 논의한 것과 같이 동사는 논항 정보를 함께 이용하여 대체 데이터를 찾아야 하지만 대부분의 온톨로지에서는 논항의 의미를 반영할 수 있을 정도로 충분한 정보를 제공하지 않는다. 미국 Colorado 대학에서 서비스하고 있는 Unified Verb Index는 VerbNet, PropBank, FrameNet, WordNet의 개념들을 연결하여 주는 프로젝트인데, 이러한 서비스를 활용하는 것도 하나의 대안으로 보인다[7].

(4) 어떤 단어에 해당하는 애니메이션 모델이 여러 개

있을 때, 이 중에서 어떤 것을 선택해야 하는가 하는 문제의 해답이 되지 못한다. 예를 들면 boy에 해당하는 모델이 두 개 존재하고, 그 중 하나는 6세 남자 어린이의 모습이고, 나머지 하나는 17세 청소년의 모습일 때, 이 중 어느 것을 선택하는가 하는 것은 해당 단어가 쓰이는 문맥에 따라 달라질 것이다. 이를 구분해주기 위해서는 문맥에서 추출해야 하는 정보들을 미리 정의하고, 각 모델에 해당 정보들을 미리 태깅(tagging)해 두는 것이 필요하다.

6. 결론

지금까지의 텍스트 애니메이션 관련 연구들은 의미를 분석하고 시각화하는 과정에 초점을 맞추고 있어, 시스템이 실생활에서 활용되기에는 너무 한정된 단어만을 다룬다는 한계를 가지고 있었다. 본 논문에서는 특히 명사를 대상으로 하여, 온톨로지의 계층정보를 참고하여 모델 DB에 제공되는 데이터 중 의미를 제대로 전달할 수 있는 모델을 선택하는 방법을 논의하였다. 향후 연구 범위를 동사로 확장하여, 애니메이션에서 적합한 동작을 찾는 문제에 대해서도 논의할 예정이다.

Acknowledgements

본 연구는 첨단정보기술연구센터를 통하여 한국재단의 지원을 받았음.

참고문헌

[1] Kyung Wha Hong and Jong C. Park, “Anaphora Resolution in Text Animation”, Proc. the IASTED International Conference on Artificial Intelligence and Applications, pp. 347~352, Innsbruck, Austria, 2004.

[2] Bob Coyne and Richard Sproat, “WordsEye: An automatic text-to-scene conversion system”, Proc. SIGGRAPH, pp. 487~496, LA, USA, 2001.

[3] Richard Johansson, David Williams, Anders Berglund, and Pierre Nugues, “Carsim: A system to visualize written road accident reports as animated 3D scenes”, Proc. the Second Workshop on Text Meaning and Interpretation, 42nd Annual Meeting of the Association of Computational Linguistics, 2004.

[4] Zhi-Qiang Liu and Ka-Ming Leung, “Script visualization (ScriptViz): a smart system that makes writing fun”, Soft Computing - A Fusion of Foundations, Methodologies and Applications, Vol. 10, No. 1, pp. 34~40, 2006.

[5] Minhua Ma and Paul Mc Kevitt, “Visual Semantics and Ontology of Eventive Verbs”, Natural Language Processing - IJCNLP-04, First International Joint Conference, pp.187-196, Hainan Island, China, March 22-24, 2004.

[6] http://wordnet.princeton.edu/

[7] http://verbs.colorado.edu/verb-index/