

GPG v0.8 사용설명서

저작권자	김동호 (DYMATION@naver.com)	작성일자	20060924
------	--------------------------	------	----------

1. 개요

1.1. Gear Profile Generator = GPG

기어 형상 작도용 프로그램 (프리웨어)

1.2. 히스토리

개정날짜	버전	개정내용
20050425	V0.5	연산, 입출력 기능 구현
20050426	V0.6	그래프상에 설명 문자 표시 추가. HELP 창의 크기 변경 기능 버그 수정. 파라미터 파일 읽어들이 때 디그리-라디안 변환 관계 버그 수정.
20050427	V0.7	화면 표시 Radius of Pitch 설명 문자 오류 수정. *.scr 스크립트 버전 표시 오류 수정.
20060730	V0.8	파라미터 입력 한계를 재조정. 파라미터 표시 정밀도를 소숫점 8자리 까지 올림. 외경 및 내경 원의 표시 오류 수정 결과 스크립트 파일 출력시 압력각 표기. 오류 수정전위계수 줄 경우에 피치원, 외경원,내경원 표시 오류 수정.

1.3. 작성 동기

저는 현역 설계 기술자로서, 설계 작업 중에 간혹, 간이 기어 형상을 사용할 필요가 생기곤 했습니다.

물론 일반적인 기계 시스템 설계에서는 기어의 이빨 형상까지 완벽하게 그려낼 필요는 전혀 없고, 필요한 기어의 사양을 간단히 도면에 표시해 주면 됩니다.

그러면 기어 전문 제작자가 주어진 사양에 맞추어 기어를 깎아서 만들어 주니까요.

프린터 같은 기계에서도 기어가 많이 사용되는데, 현역 프린터 기구설계자들 역시 플라스틱 성형 기어를 사용하기는 하지만 자신들은 단지 피치원 직경이나 기타 간단한 요소표만을 지정해 줄 뿐이라고 하더군요. 프린터 완성품 제조업체는 중견기업 이상의 규모를 갖춘 경우가 많고, 아이템 자체가 업체 간 협업 시스템이 잘 구축되어 있기 때문에, 전통적으로 오랫동안 프린터 기어만 전문적으로 공급해온 협력업체와 이런 사항이 잘 합의되어 있는지라, 기어 전문 업체에서 형상 설계 및 금형 제작, 부품 공급까지 모두 책임져 주니까요.

하지만 그런 시스템에 포함되어 있지 못한 경우라면, 부득이하게 자신이 필요한 기어를 스스로 그릴 수 밖에 없을 것입니다.

특히 호빙머신 등으로 깎아서 만드는 기어의 경우에는 상관없겠지만, 금형의 형상을 정확히 정의해 주어야 하는 플라스틱 성형 기어의 경우에는 기구설계자가 스스로 자신의 기어 형상을 모델링하거나 작도해야 할 것입니다.

또한 자신이 만일 FEM 해석을 하는 엔지니어라면, 기어의 강도 등을 해석하기 위해 정확한 기어의 형상 데이터가 필요할 것입니다. 기어 샘플을 가지고 3차원 스캐닝 따위의 야단법석을 떠는 것은 정말 낭비잖아요.

이럴 때 GPG 프로그램을 적절히 사용해 볼 수 있다고 생각합니다.

기어 형상의 작도 방법은 해석적인 방법(좌표를 함수로 만들어 내는 것)으로 그리는 것이 가장 효과적이라고 합니다. 컴퍼스와 삼각자로 그리는 그래피컬한 방법은 기어 형상 작도에서 별로 도움이 안되는 것 같습니다.

왜냐면 가장 일반적으로 사용하는 기어 형상을 표현해 주는 인볼류트 곡선은 함수 형태로도 표현되기 때문이죠. 인볼류트 곡선의 정의 그대로 정말 실패에 실을 감아서 끝에 샤프심을 꽂아놓고 빙빙 돌려가면서 그리는 사람은 없을 테니까요. 결론적으로 기어 형상 작도를 위해서는 컴퓨터 프로그램으로 작성해서 좌표 데이터를 뽑아내서 이용하는게 제일 낫다는 것이죠.

물론 기존의 기어 형상 작도 프로그램들도 있습니다.

특히 류재완 기술사님께서 온라인상에 공개해 주신 프로그램이 가장 큰 영향을 주었습니다. GPG에 사용된 함수 역시 그분의 것과 동일합니다. 결과물 자체는 별 차이가 없다는 거죠. 다만, 류 기술사님께서 프로그램을 간략히 작성하신 고로 약간의 불편함이 있어서, 저는 좀 사용자가 편하게 기어를 그리고, 몇 가지 옵션을 좀 더 조정할 수 있도록 나름대로 기능상 개선을 시도해 보았을 뿐입니다.

저도 전문적인 프로그래머가 아니기 때문에, 프로그램 작성 능력은 전혀 뛰어나지 않으며 기본적인 C 언어 정도를 사용할 뿐입니다. 다행히 학창시절에 제어공학 실험실에서 좀 빌붙어 있던 관계로 알게된 미국의 National Instrument사의 LabWindow/CVI 라는 개발 도구를 사용할 수 있었습니다.

이 개발환경은, C 및 C++을 사용하여 자동제어 또는 데이터 수집용 어플리케이션을 PC 기반으로 편하게 작성할 수 있도록 개발환경의 유저 인터페이스가 아주 쉽게 되어 있고, GUI 작성도 드래그&드롭으로만 가능하게 도구가 지원되며, 관련된 많은 라이브러리를 지원해 주기 때문에 저와 같은 사람에게 아주 고마운 도구인 것 같습니다.

원래는 GPG와 같은 성격의 프로그램 작성 용도가 아니고, PC에 연결된 계측기나 PLC 설비 따위를 제어하는 프로그램을 작성하기 위한 목적의 개발도구이지만, GPG 코딩에서도 유용하게 써먹을 수 있었습니다.

아무튼, '쉽게쉽게' 기어 형상 작도 문제를 해결코자 하는 학생이나 기술자에게 약간의 도움이 된다면 저로서는 아주 기쁘겠습니다. 이 문서 역시 그런 도움을 드리기 위한 것입니다.

* GPG v0.8을 다운로드 받을 수 있는 곳 : <http://blog.naver.com/dymaxion/15000700095>

2. 기어 소사(小史)

제가 역사 매니아인 이유로 기어 역사에 관해서 간단히 검색해 본 걸 정리해 봤습니다. 뚝금없다고 하지 마시고 그냥 넘어가 주시길...

인류가 기어의 원리를 자각하게 된 시점은?

아주 오래 되었다고 합니다.

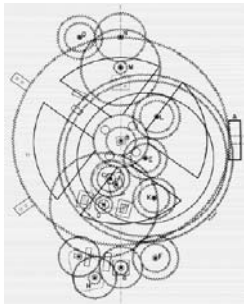
일반적으로 BC 20세기를 전후해서 그 시작으로 보는 것 같은데요. 근거는 당시의 토기나 장신구 같은 유물들을 보고 분석해 보면, 기어의 원리를 이용한 도구를 사용하여 반복적인 무늬 같은걸 넣은 것으로 추정되는 경우가 있기 때문이라고 합니다. 뭐 혹시 아니까?!

한국의 빗살무늬토기의 빗살무늬 넣을때도 기어의 원리를 이용한 도구를 사용해서 작업했을지도...

BC 4세기에는 아리스토텔레스가 기어를 적극적으로 사용하고 있었던 것은 확실하고, 아리스토텔레스가 저술했다는 '기계 문제(원계는 모름)'라는 책도 있었다고 하는데요. 그러니깐 이때쯤엔 인류 문명의 최첨단 지역에선 최소한 기어를 제대로 이용하고 있었다고 봐도 될 것 같습니다.

BC 3세기에는 아르키메데스가 이미 웜기어를 적용하고 있었다고 합니다. 그리스 신화에 나온 천문 시물레이션 시계의 실존을 증명하는 유물도 나왔다고 하니.. 대단합니다.

(1901년 펠로폰네소스 반도 남단 안티키테라 섬 앞바다의 로마 시대 난파선에서 에서 잠수부들이 인양함)



↑ 그리스 유물의 복원 구조.

복잡한 기어 장치로 천문 12궁의 움직임을 재현



↑ 중국 지남차 상상 복원품.

황제가 치우와 전쟁할 때 써먹었다고 하지만, 진짜인지 구라인지...

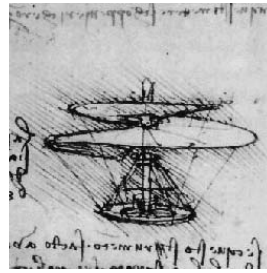
그리스,이오니아 문명이 정말 대단했던 것은 부정할 수 없다고 생각합니다.

하지만 중세 시대에는 사실 공학적인 부분에서 거의 진보가 있기는 커녕 퇴보하거나 않았으면 다행인 것은 사실이구...

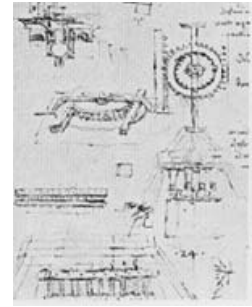
동양 쪽에선 기어 같은걸 제대로 이용한게 지남차 같은 전설적인 풍문 이외에는 별로 눈에 띄질 않아 안타깝습니다.

15세기엔 레오나르도 다빈치가 헬리콥터의 아이디어도 스케치하고 등등 거기서 나무로 조립해 만든 기어 구조가 보이는데 혁신적이고 대단한 아이디어 이지만, 순수하게 기어 디자인 측면에서두 대단한 발명이 이루어진 거 같아요.

공식적으로는 나사기어, 베벨기어, 웜기어, 하이포이드 기어 등이 레오나르도 다빈치 발명으로 인정받고 있는 것 같습니다.



↑ 레오나르도 다빈치 헬리콥터



↑ 레오나르도 다빈치 동작기계

서양에서 제대로 기어 디자인에 대해 관심을 가지기 시작한 건... 역시 '슈퍼 울트라 만능 완벽 무결점 초천재 수학자'인 오일러(Leonhard Euler) 인 것 같아요. '인볼류트 함수'란 것도 오일러가 만들어 낸 것이라고 합니다. 여기서 잠깐 제가 학교 땡길때 수업 중간 휴식시간에 복학생들이 안돌아가는 머리로 수업 따라가기 힘들어서 담배 백백 피우면서 서로 했던 반농담이 생각납니다.

"야, 타임머신이 발명되면 암살대상 1호는?"

"오일러를 죽여야 해." (-_-);

오일러만 없었다면 기계 문명이 이렇게 극단적으로 발달되지도 않았을테구, 우리같은 공학도들이 오일러 때문에 골머리를 썩히지도 않았을거 아니냐는 푸념이었는데..

제 기억에 고등학교 수학에서도 오일러란 이름이 심심찮게 나오죠??? 과연 오일러는 기계공학 과목의 전과목의 교과서를 장식하는 진짜 울트라 초(超)천재 같습니다. 유체역학에 오일러리안 나오는데 독보적인 방법론으로 뉴턴 방법론이랑 쌍벽 이루고. 고체역학(재료역학)에 오일러식 나오고. 기초중의 기초. 제어공학에 오일러 방정식 풀이 방법 나오고. 고당때 배우지만 미분방정식 풀때는 역시 오일러 오일러 오일러.. 기구학에도 이런 기어 과정에 오일러 나오고. 도대체 오일러란 이름이 안나오는 과목이 없어요.



↑ 오일러 (1707~1783)

인볼류트 곡선의 우월성 주장.



↑ 데자르그 (1591~1661)

외(外)사이클로이드 함수를 발명.

서양에선 사이클로이드 곡선을 사용한 기어가 기원전부터 있었다는 이야기가 보이는데 확실한 증거는 제가 잘 모릅니다.

17세기 부터는 사이클로이드 기어에 대한 연구가 확실히 있었던 거 같아요. 이때 불란서의 '데자르그'라는 수학자의 경우엔 기존의 사이클로이드 곡선을 개선한 '외(外) 사이클로이드 곡선'을 개발해서 더욱 마찰이 적고 마모가 줄어들어 수명이 개선되는 효과를 가진 기어를 개발했다고 합니다.

1733년에는 역시 불란서의 '까무'라는 엔지니어는 '사이클로이드 기어가 기어의 형상으로 이론상 최고로 완벽하다'라는 주장을 근거화 해냅니다.

하지만 '슈퍼 울트라 초천재' 오일러 때문에 갑자기 인볼류트 곡선을 사용하는 기어로, 트렌드 자체가 확 바뀌어 버립니다. 오일러는 1754년에 사이클로이드 보다 인볼류트 기어가 더 낫다고 주장하는 논문을 씁니다.

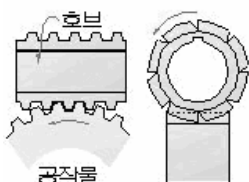
1771년에는 '케스트너'라는 사람이 역시 오일러의 주장을 뒷받침하는 새로운 근거를 더욱 많이 발굴해서 논문을 씁니다.

잘 만들 경우(실제 제작하는 테크니션의 실력이 엄청 좋아서 제대로 만들 경우에) 사이클로이드 기어가 더욱 성능은 좋지만, 더 까다롭고 만들기도 힘들고 표준화하기도 어렵기 때문인데요. 인볼류트 기어의 경우엔 모듈만 대충 맞으면 잘 물려 돌아가거든요. 심지어 모듈 파라미터가 달라도 압력각 파라미터만 대충 잘 계산해서 맞추면 잘 맞아 돌아가는 신기하게 털털한 성질이 있는 신비로운 곡선이라서 그렇습니다. 공차 허용 범위도 더 넓고요.

18세기에는 내연기관이 등장... 고속 대출력 회전 동력 소스가 나타남으로 인해서, 기어의 필요성이 본격적으로 대두되고. 기어에 관심을 가지는 사람들도 많아진 것 같아요.

19세기 중반에 들어오면 서양에서 산업혁명이다 과학혁명이다 뭐다 해서 난리 법석 떨면서... 순진하게 잘 사는 남의 나라 침략도 해 가면서 지구를 못살게 구는데요.

1900년에는 기어를 깎는 대표적인 기계로 오늘날에도 썩썩하게 돌아가는 '호빙 머신'이란 녀이 발명됩니다. 독일에 '헤르만 파우터(Herman Pfauter)' 라는 엔지니어가 만든 거랍니다.



↑ 호빙 머신



↑ 제임스 글리슨
글리슨-파우터사 CEO.
파우터의 호빙 머신을 개량한 글리슨의 후손으로 생각됨.

기어를 만드는 방법은 여러가지가 있는데, 그중에 호빙 방법이 제일 일반적이며 청계천이나 구로동 독산동 쪽에 자영업자들이 간판에 '기어 제작'이라고 해 놓았다면 열에 아홉은 호빙 머신 하나씩 갖다놓고 돌리고 있을 겁니다.

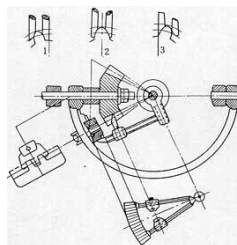
사실 이 기계는 원리 자체는 별로 특별한 것은 없으나, 교묘하게 안배된 공구와 절삭물간의 회전수 비율에 의해서 특정한 사양의 기어 곡면을 깎아낼 수 있다는 점이 신기하죠.

항공기용 기어의 경우엔 사용연한 2년이 채 되지 않은 신형 고급 호빙 머신으로만 깎은 걸 사용하는게 일반적이라고 합니다. 공작기계란 것이 역시, 사용 기간이 길어지면 오차도 커지게 마련이니까요.

기어의 강도와 신뢰성을 높이는 제대로 된 시도들은, 두 번의 세계대전을 거치면서 프레데릭 풀러 등의 전략가들이 '기계화 전쟁' 개념을 확립하면서, 기계의 중요성이 전쟁에서 본격적으로 대두되면서 서양 각국이 미친듯이 개발한 결과일 텐데요.... 기어를 깎고 나서 마무리를 어떻게 하나, 열처리를 어떻게 하나 등등의 세부적이고 구체적인 기술들은 많은 이름 없는, 일에 미친 엔지니어들에 의해서 이루어 졌습니다.

엔지니어의 일이란... 천재적인 영감에 의해서 획기적인 노력이션을 슬쩍 연습장 한 장에 써내려가서 업적을 남기는 경우는 희귀하고, 대부분은 끝없는 '노가다'의 결과인 것 같아요.

20세기 초엽에 들어오면서 인볼류트 기어 이외의 더 좋은 형상은 없을까 많은 연구가 이루어진 것 같은데... 예를 들면 영국의 '프랭크 험프리스'라는 엔지니어가 '원호 치형'이란 것은 개발한 것. 이후 1912년에 역시 영국의 '스미스'란 엔지니어가 원호 치형을 깎아서 제작하는 기법을 포함한 솔루션을 제시했구요. 1958년에는 소련의 '노비코프'라는 엔지니어가 역시 기존의 원호 치형을 개선해서 인볼류트 기어보다 훨씬 큰 부하를 견딜 수 있는 기어의 형상을 설계하는 방법을 제시했다고 합니다. 노비코프 기어의 경우에는 형상이 희안하게도 기존의 인볼류트 기어가 꼽아졌던 자리에, 새로 깎은 노비코프 기어를 꼽아도 호환이 가능하다는 오묘한 결과가 나온다고 하네요.



↑ 글리슨식 베벨 왕성 구조



↑ 글리슨사의 40시리즈 스파이럴-베벨 기어 플래너

스파이럴 베벨기어의 경우, 글리슨(Gleason)사에서는 원호치형을 자사 표준으로 하고, 이태리의 피아트(Fiat)사에서는 사이클로이드 곡선을 표준으로 쓰고 있고, 독일의 킹겔른베르크(Klingelberg)사에서는 인볼류트 곡선을 표준으로 삼고 있다고 하네요. 시장 지배력은 글리슨사가 월등한 것 같습니다.

20세기에 들어와서 이런 기어들에 대한 체계적인 정리가 이루어지고 좋은 책들도 많이 나오는데.... 지금도 기어 공부하는 사람들에게 최고의 베스트셀러이며 바이블이 된다는 "더들리의 기어 핸드북"을 쓴 더들리 아자씨.

기어의 이빨 형상의 성능을 평가하는 해석 기법을 체계적으로 제대로 제시한 MIT의 버킹엄 교수.

등등의 인물들이 표면적으로 나타나는 듯 합니다.

하지만 이외에도 잘 알려지지 않은 많은 훌륭한 엔지니어들이 있겠죠?

우리나라의 경우엔 1950년대, 한국전쟁 직후에 기어를 깎아서 만드는 한국 고유의 최초 업체가 등장한다고 하는데, 그 회사는 지금도 현존하는 것 같아요. 불행히도 상호는 제가 기억이 잘 안납니다. 하지만 신기하지 않습니까?

한국의 허접해 보여도 의외로 이런 부분에서는 놀라게 하는 부분이 있습니다. 한국의 테크니션, 엔지니어 선배님들 정말로 존경합니다.

글리슨사 같은 회사가 소위 말하는 '원천기술'이란 걸 가진 거겠죠. 우리나라 회사 중에서도 저런 지위를 가진, 공학 자체를 선도하는 진짜 원천기술을 보유한 업체들이 많아지면 좋겠습니다. 이런 '진짜 기술'에는 정치가나 관료들의 협잡이나 돈 놀이 따위가 끼어들 여지가 없는, 진정한 엔지니어링으로 승부를 봐야 한다고 생각합니다.

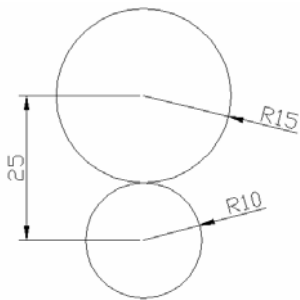
3. GPG를 위한 간략 기구학

GPG를 사용하는데 도움이 되는 정도의 간단한 개념만 정리해 보겠습니다. 깊게 들어가면 저도 모르고, 완전 학문의 세계가 되기 때문에...

고수(高手)분께서는 보시고 한 번 헛헛 웃어 주시고, 내용상 오류나 보완점이 눈에 띄신다면 저에게 메일을 보내 주십시오. 반영해서 업데이트 하겠습니다.

3.1. 피치원(Pitch Circle)

우선 가장 기본이 되는 2개짜리 평기어를 설계한다고 가정해 봅시다. 그러면 상식적으로 원을 두 개 그리고, 그 두 원이 접하도록(Tangential) 배치를 하게 되겠죠. 이게 '피치원'입니다.



↑ 피치원 작도에 (중심거리 25mm, 기어비 3:2)

피치원을 그리면 기본적인 기어 스펙이 나오죠.

중심 거리가 나오고, 기어비가 나옵니다. 또, 각 기어의 반경도 나옵니다.

이정도면 일단 설계할 때 레이아웃을 잡을 수 있고, 또 기어비가 나오기 때문에 입력단과 출력단의 속도 차이 같은 것도 나오니까 기구적인 핵심은 이미 해결이 되었잖아요?

3.2. 잇수(Number of Teeth, N)

기어비가 나오면 각 기어의 잇수를 선정할 수 있을 겁니다.

반경 15mm : 10mm 기어 조합의 경우, 그냥 숫자대로 15개:10개 요렇게 맞춰도 일단은 큰 문제가 없겠죠?

아니면 이 비율을 그대로 늘였다 줄였다 하면서 잇수를 조정할 수도 있겠어요. 예를 들면 150개:100개 라든지, 30개:20개 이런 식으로요.

물론 고속용이나 대하중용 기어의 경우에는, 이런 잇수를 선정할때도 제한조건 같은게 여러가지 붙습니다. 예를들면 앞서 예를 든 15mm : 10mm 비율의 경우 별로 좋지 않을 가능성이 높습니다. 왜냐면 기어비가 정수로 딱 떨어지기 때문이죠. 이렇게 정수로 딱 떨어지는(즉 최대공약수로 떨어지는) 경우에는, 기어가 계속 회전할 때 한 번 맞물렸던 특정한 이빨이 몇번 회전하면 또 다시 만나는 상황이 연출되기 때문에 안좋다고 합니다.

한 번 만났던 이빨끼리는 가급적 다시 안만나도록 하고, 다른 이빨들과 골고루 만나주도록 하는게 좋다는 건데요. 왜냐면 기어 이빨들이 서로서로 평균화되게 골고루 잘 길이 들 수 있도록 배려하기 위해서라고 합니다.

하지만, 장난감 같은데 쓰는 간단한 간이 기어박스 같은데서는 이런 제한사항에 굳이 얽매일 필요는 없겠죠. 시스템의 신뢰성이나 사용조건이 극단적인 경우는 아니니까요.

아무튼, 피치원과 잇수 사이에는 모듈(Module) 팩터가 붙어서 관계를 이루죠?

3.3. 모듈(Module, M)

가장 기본적인 관계식이고, '모듈'이란 팩터는 기어간의 호환성이나 표준화 따위를 위한 기준으로 삼기 위해 도입한 개념 같습니다.

$$D_p = M \times N$$

여기서 D_p 는 피치원 직경(Diameter)입니다.

피치원 직경 30mm에 잇수 60개짜리 기어라면 모듈이 0.5이라는 식이겠죠.

기어 전문 회사에서 나오는 표준 기어의 경우엔 주로 쓰는 모듈들이 도표로 정리되어 있어서, 그걸 골라 쓰면 되지만...

GPG에서는 0.8946 따위의 모듈도 그럴 수 있습니다.

모듈이 이런 식으로 세밀하게 조정되어야 할 경우도 있으니까요.

3.4. 압력각(Pressure Angle, ϕ)

두 기어의 공통 접선과 작용선 사이의 각도를 압력각이라고 하는데, 작용선이란건 별게 아니고 두개의 베이스원을 동시에 접하도록 지나가는 선을 말하죠?

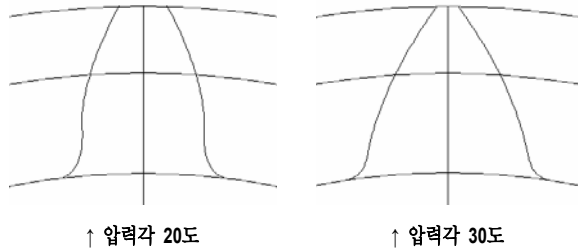
제일 많이 쓰는 표준 압력각은 20도 라고 하고요.

* 베이스원 : 실패에 실이 풀리는 인블류트 곡선에서 실패에 해당.

따라서 압력각이 커진다는 것은 베이스원이 작아진다는 의미와 같고, 베이스원이 작아진다는 것은 실이 풀려나오는 실패가 작아지는 거니깐, 인볼류트 곡선이 안쪽 깊은 곳에서부터 시작된다는 것이고, 때문에 바깥으로 풀려 나오면 나올수록 더 많이 좁아져 있을 것으로 감잡을 수 있습니다.

한마디로 압력각을 키우면 기어의 꼭대기가 좁아집니다.

GPG로 압력각을 디폴트 20도로 놓고 그려본 다음에, 압력각을 바꿔가면서 형상의 변화를 확인해 보는 것도 좋을 것입니다.



압력각 변화로 발생한 형상의 변화는, 기어의 물림률 같은 것에도 영향을 주지만 여기서는 따지지 않겠습니다. 나중에 혼자 천천히 공부해 보세요~ (무책임)

3.5. 어텐덤(Addendum)과 디텐덤(Dedendum)

이건 쉽게 말해 기어의 이끝 직경과 이뿌리 직경을 표현한 것입니다. 다만 피치원을 기준으로 올라온 높이와 낮아진 높이를 따진 것이어서 이런 용어를 쓰는 것 같아요.

피치원 반경(Radius)이 20mm 일 경우에, 어텐덤이 3mm라면 기어의 이끝원 반경이 23mm 라는 소리이고, 디텐덤이 3.5mm 라면 이뿌리원 반경이 16.5mm 라는 것이겠죠.

어텐덤과 디텐덤을 이처럼 실측 단위로 표현하기도 하지만, 어떤데서는 비율(Ratio)로 따질 경우도 있던데요. 피치원 반경 기준으로 디텐덤 0.9 라든가, 어텐덤 1.05 라든가 말이죠.

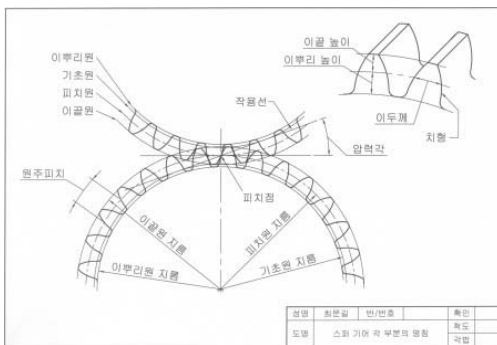
GPG 프로그램에서는 이런 비율이 아니라 실측 단위를 그냥 써넣도록 했습니다. 사용자 입장에서는 실측 수치로 넣는게 더 편한 것 같아서요.

단, 절대로 지켜야 할 원칙은 이것입니다.

$$\text{Addendum} < \text{Dedendum}$$

왜냐? 기어가 돌아갈 때 간섭을 피하기 위해서 입니다.

인터널 기어(Internal Gear)로 형상을 이용할 경우에는, 이것이 반대가 됩니다.



↑ 기어 기구 용어 (피운 것임)

3.6. 전위량(Offset Factor, X)

전위량을 0인 것을 표준기어라 하고, 원래 정해진 피치원 직경이 그대로 유지됩니다.

하지만 전위량을 줄 경우에는 피치원 직경이 변화합니다. 전위량을 양수로 주면 피치원 직경이 좀 더 커지고, 음수로 주면 피치원 직경이 작아집니다.

한마디로 이걸 바꾸면 기어가 커졌다 작아졌다 합니다.

때문에 기어의 중심거리가 바뀌겠죠?

하지만 기어의 모듈과 잇수는 유지되기 때문에 호환성은 그대로 유지됩니다.

GPG 프로그램에서 전위량은 어텐덤 수치에 대한 비율을 써넣도록 되어 있습니다.

어텐덤이 2mm일 때,

전위량을 0.5로 주면 피치원 직경이 1mm 커집니다. 물론 기어의 형상 자체도 그만큼 전위(오프셋) 됩니다. 기어가 약간 커지죠.

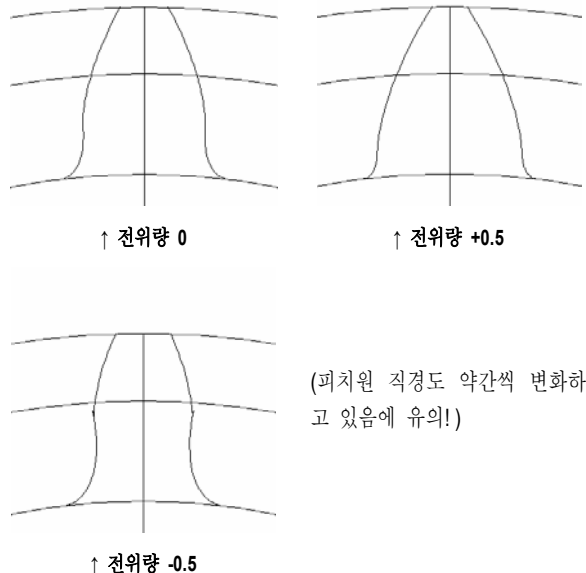
전위량을 -1로 주면 피치원 직경이 2mm 더 작아집니다. 기어가 작아집니다.

이렇게 전위가 되면서, 기어의 이빨 형상도 약간씩 바뀝니다.

양수로 전위될때는 기어의 이끝이 좁아집니다. 기어가 날카로와지죠. 대신에 이뿌리 쪽은 두툼하게 보강됩니다.

음수로 전위될때는 기어의 이끝이 두툼해집니다.

두툼한 기어가 더 튼튼할 테니, 음수로 주는게 언뜻 좋아 보입니다. 하지만, 기어 끝이 두툼해지는 대신에 기어 이뿌리 쪽으로는 이빨이 콜라병처럼 잘록해 집니다. 잘록한 기어는 언더컷이라고 해서 웬만해선 피하는게 상책이죠.



피치원 직경이 이렇게 변화되기 때문에, 기어의 중심거리가 바뀌게 되니깐 설계할 때 상당히 짜증날 것입니다.

하지만 굳이 이걸 써야할 때가 있다고 합니다.

두가지 경우입니다.

1. 기어 이빨의 언더컷 형상을 개선하고 싶다.
2. 기어 중심거리와 각 기어의 반경이 약간 안맞아서 정확히 맞춰주고 싶다.

1번 경우에는, 압력각을 바꿔줘도 어느정도 해결 가능하지만 이처럼 전위량을 조정해서 해결할 수도 있겠죠. 물론 언더컷을 없애려면 양수로 줍니다. 그렇다고 너무 큰 숫자를 주면 기어의 이끝이 바늘처럼 날카로와 지거나 그걸 넘어서서 아예 파먹는 경우가 생기니깐, 적당한 선에서 타협해 줄 수 있습니다.

2번 경우에는, 예를들어 이런 경우를 상정해 볼 수 있습니다. 기어간 중심거리가 25.5mm로 이미 정해져 버린 상황인데, 기어의 피치원 반경이 각각 15mm, 10mm 인 경우.

0.5mm 벌어지게 되므로 이걸 보정해 줘야 될 상황이 생길 수 있습니다.

이때 전위량을 살짝 줘서 양쪽 기어의 피치원 반경을 0.25mm 씩 늘려주면 문제가 해결되겠죠?

(현재의 GPG에서는 어텐덤에 대한 비율(Ratio) 값을 입력하도록 해 놓았는데, 역시 실무에서 편하게 쓰려면 아무래도, 실측 수치로 전위량을 써넣도록 변경할까 생각중입니다.)

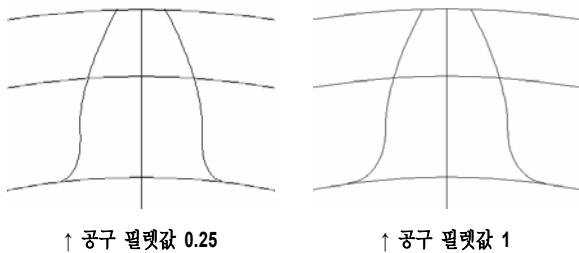
3.7. 필렛(Fillet)

인볼류트 곡선 하나만 가지고 기어를 그리면, 기어의 이뿌리 부위에 필렛(라운드)이 없기 때문에 응력이 집중되는 구석진 곳이 생깁니다. 때문에 필렛을 줘야 하는데요.

그냥 Autocad 같은데서 주는 자동 필렛 같은 것도 간섭(Interference)만 없다면 무방할 것입니다.

하지만 여전히 기어 작동시 빈공간(Dead Space)가 있을 수 있으므로 일반 필렛은 최적화 측면에서 좋다고 할 수 없고, 또한 사출성형 기어가 아니라 호브로 깎아서 만드는 기어의 경우에는 공구 날끝의 반경에 따라서 필렛값이 정해지게 되므로, 이점을 고려해서 정확히 작도해 주면 좋을 것입니다.

GPG에서는 인볼류트 곡선과 함께 트로코이드 필렛 곡선도 연결해서 작도해 주도록 되어 있는데, 이때 입력해 주는 필렛 반경(Fillet Radius)는 공구의 필렛값을 의미하므로 실제 그려지는 필렛 결과는 이보다 좀 더 반경이 커집니다.



↑ 공구 필렛값 0.25

↑ 공구 필렛값 1

GPG에서 필렛값을 키워주면 작도되는 트로코이드 필렛 곡선도 커집니다. 필렛이 커질수록 언더컷을 뭉개주는 효과(?)도 부수적으로 발생하는데 그렇다고 너무 키워주면 이빨형상이 괴상하게 변하니깐 적당히 맞춰주면 될 것이고...

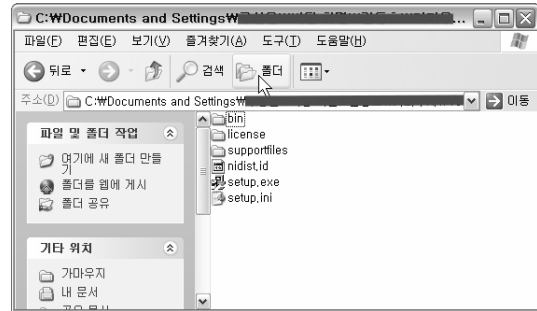
이빨 형상 자체는 큰 문제가 없더라도, 너무 큰 필렛값을 써서 작도하면 이웃한 이빨에서 나온 필렛간에 서로 겹쳐서, 이뿌리원을 완전히 덮어버리는 수도 생깁니다.

이런 부분을 별도로 검증해 주는 기능은 GPG에 없으므로, 몇번 반복해서 작도하면서 최적의 형상을 찾도록 하면 될 것입니다.

4. GPG 깔기

4.1. 인스톨 준비

첫페이지에 소개된 URL은 저의 개인 블로그입니다. 여기서 분할 압축된 인스톨 파일을 다운로드 받을 수 있습니다. 압축을 풀면 다음 파일들이 나옵니다.



↑ GPG v0.8의 인스톨 준비

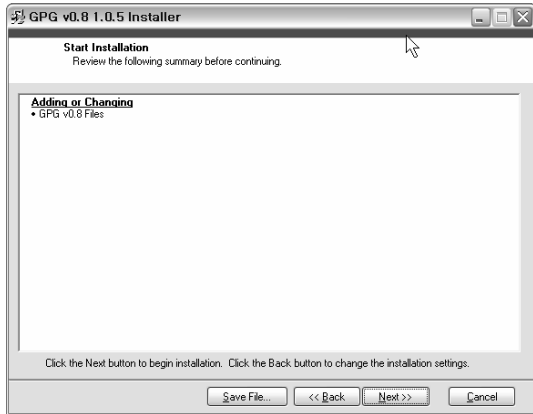
여기서 Setup.exe를 더블클릭해서 실행하면 인스톨이 시작됩니다.

4.2. 인스톨



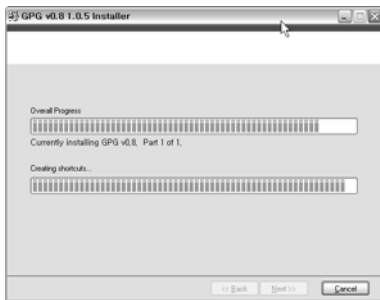
↑ GPG v0.8 인스톨 위치 지정

인스톨러가 시작되면 먼저 인스톨될 위치를 지정하는게 나오는데, 적당한 위치로 새로 지정해 주거나 그냥 디폴트 디렉토리로 놔두거나 알아서 하세요~



↑ 그냥 Next 누르세요.

다음으로 넘어가면 위와 같은 창이 나오는데, 그냥 Next 누르세요.



↑ 인스톨 진행

다음에는 위와 같이 인스톨이 진행됩니다. 작은 프로그램이므로 금방 끝날 것입니다.



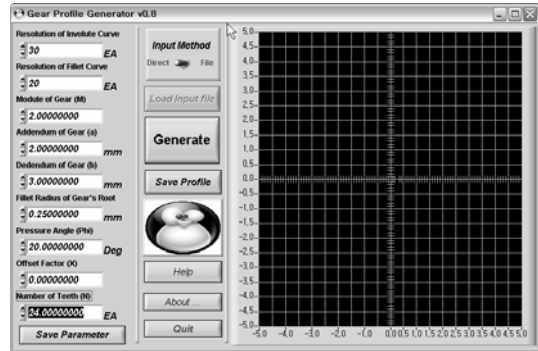
↑ 인스톨 완료

이제 완료되었으므로 Finish를 눌러 줍니다. GPG의 실행 아이콘은 시작→프로그램 메뉴에만 등록되도록 해 놓았습니다. 바탕화면에는 아이콘이 생기지 않습니다.



↑ 프로그램 메뉴에서 실행 아이콘 확인

5. 사용법



↑ 처음 실행 했을 때의 상태

프로그램을 시작하면 위와 같은 창이 나옵니다. 좌측의 파라미터들을 입력해 주고 나서 'Generate' 단추를 누르면 우측의 그래프에 그림이 나옵니다.

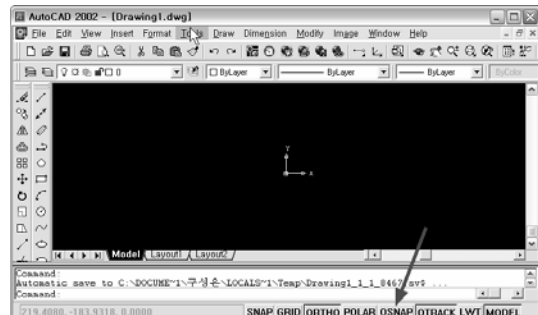
입력한 파라미터들을 잊어버릴까 걱정된다면, 'Save Parameter' 단추를 눌러서 저장해 주시면 됩니다. 이렇게 저장한 파라미터값을 나중에 다시 읽어들이고 싶다면, 'Input Method' 토글 버튼을 'Direct'에서 'File'로 바꾼 다음에 'Load Input File' 단추를 누릅니다. 그러면 파라미터 파일에서 읽어들이는 파라미터 값들이 적용됩니다. 물론 'Generate' 단추를 누르면 그 값대로 다시 작도되겠죠.

파라미터 파일은 *.dat 파일로 저장되고, 나중에 읽어들이때도 *.dat 확장자 파일이어야 합니다. 파일 자체는 일반 텍스트 파일이므로, 메모장 등으로 열어서 직접 수정해 줄 수도 있습니다. 다만 확장자가 바뀌면 GPG에서 못 읽습니다.

Generate된 기어의 형상 데이터를 저장하려면 'Save Profile' 단추를 눌러서 저장해 주면 됩니다.

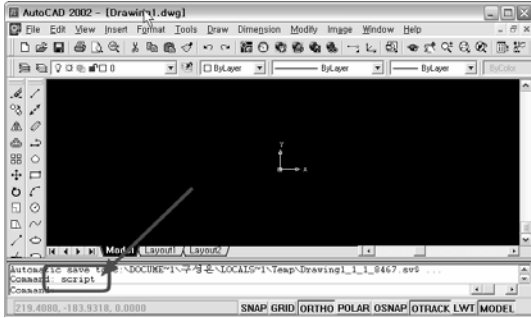
*.scr 확장자로 저장이 되며, 이것 역시 일반 텍스트 파일입니다. 다만, AutoCAD에서 실행 가능한 스크립트 파일입니다.

AutoCAD상에서 스크립트를 실행할 때, 한가지만 주의하시면 됩니다. 즉 OSNAP을 비활성화한 상태에서 스크립트를 실행해야 합니다. 그렇지 않으면 그림이 괴상하게 그려집니다.



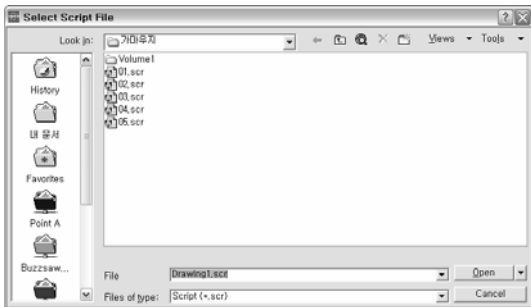
↑ OSNAP 단추를 비활성화 상태로 해 두세요.

그 다음에 AutoCAD에 'script'라고 명령을 쓰고 엔터~!



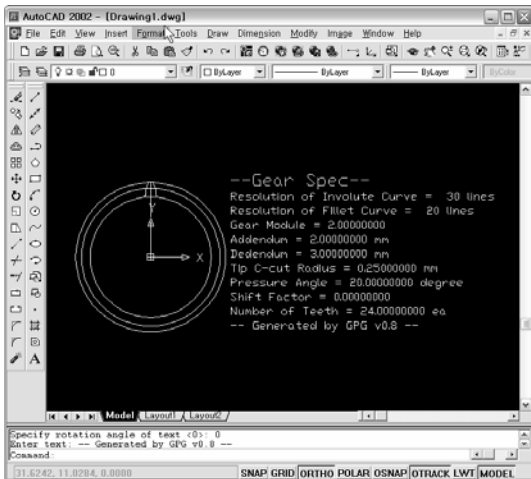
↑ script 명령을 쳐세요.

그러면 스크립트 파일을 고르는 팝업창이 뜹니다.



↑ GPG에서 저장했던 *.scr 파일을 고르고 Open.

앞서 GPG에서 저장했던 *.scr 파일을 찾아서 골라주고 Open 하면,



↑ AutoCAD 스크립트 실행 결과.

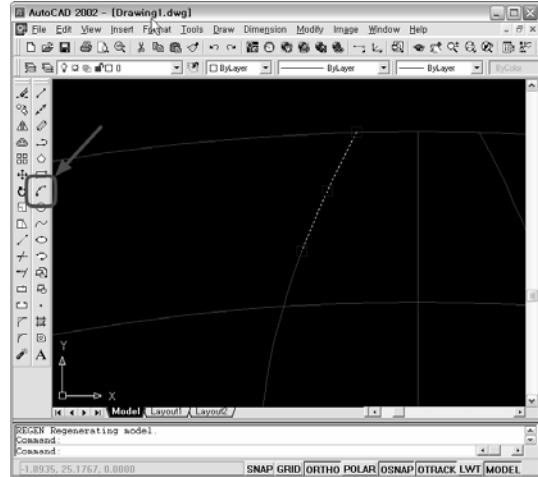
스크립트 파일이 실행되고 그 결과가 화면에 나옵니다. 기어의 이끝원, 피치원, 이뿌리원과 함께 이빨 하나의 형상이 그려져 있고.. 그 우측에 기어의 스펙이 나와 있음을 확인할 수 있습니다.

이제 이것을 이용해서 AutoCAD 상에서 지저분한 것들을 좀 정리해 주고 편집 좀 해주고 해서 원하는 기어로 설계해 들어가면 됩니다.

6. 편집하기

AutoCAD 상에서 그려진 형상을 이용하기 위해서는 약간의 편집이 필요합니다.

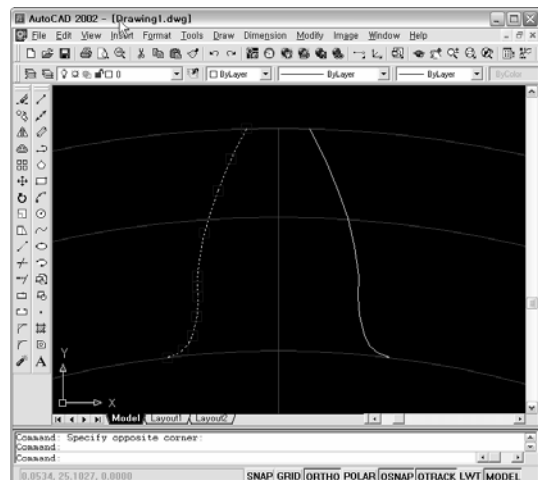
GPG에서 작도할 때 입력한 파라미터 중에서, Resolution of Involute Curve 및 Resolution of Fillet Curve에 의하여 분절되는 점의 개수가 정해지는데, 이 숫자가 높으면 좀 촘촘하게 되고 작으면 성글게 됩니다. 경험상 대충 10~20 정도가 적절한 것 같아요. 너무 많으면 편집할 때 힘드니까요.



↑ arc 명령을 이용해서 원호를 그립니다.

아까 비활성화 시켰던 OSNAP 단추를 활성화 해 주고, 위의 그림에 화살표로 지정한 arc 명령을 이용해서 원호로 기어를 따라 그려 줍니다.

대충 4~6개 정도의 원호로 기어의 형상을 근사하게 그려낼 수 있는 것 같습니다. 전부 다 그리고 나서, 원래의 분절된 선들은 다 골라서 지워줍니다.



↑ 원호로 다 그린 후에 필요 없는 원래의 분절된 선들을 다 지워준 모습.

이제 array 명령을 이용해서 기어의 잇수 만큼 빙 둘러 배열해 주면 되겠죠?

7. 연습

연습삼아 응용하는 것을 가정하여 하나 설계해 보겠습니다. '따라하기' 개념으로 읽어보시면 도움이 될 수 있을 것입니다. 물론 실전에서는 이것보다 더 많은 변수들이 고려되어야 하며 매우 조심스럽게 평가되어야 할 것입니다.

7.1. 설계 목표

7.1.1. 기어 종류 : 유성기어 박스

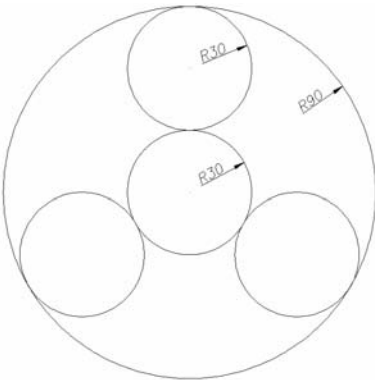
7.1.2. Pro/Engineer 2001 소프트웨어로 3D 모델링.

7.2. 기어 사양 선정

유성기어박스에서 보통 가장 작은 것은 태양기어 이니까, 태양기어의 잇수를 먼저 선정하는 것이 좋겠습니다. 기어의 잇수는 최소 10개 이상 되는 것이 언더컷 예방에 좋다고 합니다. 여기서는 15개로 선정하겠습니다.

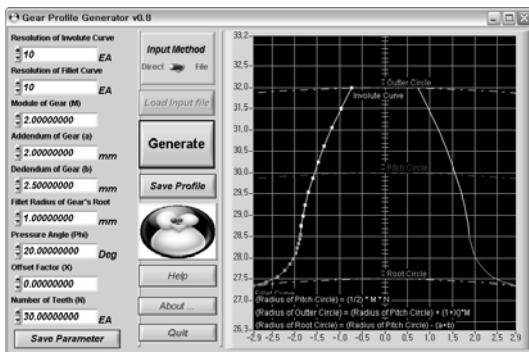
	태양기어	유성기어	링기어
피치원 반경	30mm	30mm	90mm
잇수	30개	30개	90개
모듈	2	2	2

이것은 가장 기초적인 사양이므로, 유성기어박스 설계에 관련된 여러가지 요소들을 모두 고려해서 주의깊게 선정해야 합니다. 여기서는 그런것들을 다 무시하고 그냥 간단히 임의로 정해 본 것입니다.



↑ 우선 선정된 사양을 넣어 배치해 본 유성기어박스의 레이아웃.

일단 GPG 프로그램을 시동해서, 태양기어 관련하여 위의 사양을 넣어 봅니다.



↑ 우선 선정된 사양을 넣어 그려진 태양기어의 이빨 형상.

만일 이뿌리 부위에 언더컷이 발생할 경우, 언더컷을 완화하기 위해 다음과 같은 선택을 할 수 있습니다.

1. 필렛값을 증가시켜 본다.
2. 잇수를 증가시켜 본다.
3. 압력각을 증가시켜 본다.
4. 전위량을 넣어 본다.

여기서 1번의 경우에는 기어의 특성 자체에 영향을 가장 적게 주면서도 효과가 좋은 방법이므로, 먼저 시도해 볼 만 합니다. 이것만으로 부족해서 언더컷을 완전히 없애지 못할 경우, 다음 조치 사항을 추가 적용해 봅니다..

2번 조치 사항인, 기어의 잇수를 증가시키는 것은 태양기어-유성기어-링기어 상호간에 연결된 부분이기 때문에, 하나가 바뀌면 모두가 다 바뀌어야 합니다. 잇수 관계를 잘 따져서 문제가 없다는 것이 확실한 경우에 한하여 잇수를 변경해 줄 수 있을 것입니다.

따져야 하는 잇수 관계라는 것은, 이를테면 각각의 기어의 잇수는 기어가 회전체이므로 정수로 딱 떨어져야 합니다. 기어의 잇수가 '28.3개' 따위가 될 수는 없으니까요. (그런 숫자를 채택 가능한 경우는, 기어가 한정된 범위에서 작동하고 360도 연속회전할 필요가 없는 경우에만 가능하겠죠.)

그 외에도 몇가지 사항들이 있는데 그것은 각자 공부하고 경험하시면 된다고 생각합니다.

3번 조치사항인 압력각을 변경하는 것은, 기어의 물림률 등의 요소에 영향을 주기 때문에 기어박스가 고속회전하거나 할 경우에 소음의 발생 양상이라든지 등등의 침해한 문제가 있을 경우에는 압력각을 확고히 유지하는 것에 우선순위를 더 두어야 하곤 합니다.

하지만 압력각을 살짝 늘려주면 기어 성능에 큰 영향을 주지 않으면서 상당한 효과를 볼 수 있습니다.

다만 압력각이 다르면 다른 기어와의 호환성에 관한 전망 같은 것도 예측해 보는 게 좋겠습니다.

4번 조치사항인 전위량 조절은, 피치원 반경이 변화하는 결과가 있으므로 골치가 아파게 됩니다. 하지만 실무에서 전위기어는 자주 애용되는 방법이므로 눈여겨 봐 두면 나쁠 것 없죠!

언더컷을 없앴다면, 마지막으로 기어의 이끝 부위의 폭을 한번 확인해 줍시다.

기어의 이끝이 너무 날카로와져 버리면, 이 부위가 또 약해서 문제가 생길 수 있으니까요.

예를들어 이끝 부위의 폭이 1mm가 채 못된다면 문제가 좀 있겠죠?

아무튼, 이런 식으로 눈으로 봐 가면서 사양을 조절해 가는 것은 좀 무식한 방법이긴 하지만... 의외로 효과적인 것 같기도 합니다. 파라미터 결정할 때 각종 최적화 알고리즘 같은 게 많이 있기는 하지만, 현장에서 제품 개발자가 그런 알고리즘을 구현해서 소프트웨어를 별도로 개발해 가면서 진행할 시간이 없는 경우가 많으니까요. 때로는 자신의 감(感)을 믿고 가야할 경우도 많은 것 같습니다. 물론 확신이 설 때만요!
(어떤 경우에는, 파라미터를 변경할 때 이빨이 이상하게 깨져서 형체도 알아보기 힘든 경우도 있습니다. 그럴 때는 위해서는 어텐덤 및 디텐덤 값을 조정해 가면서 보면 잡힐 확률이 높습니다.)

아튼 나머지 사양도 결정된 것 같네요.

	태양기어	유성기어	링기어
어텐덤	2.0mm	2.0mm	2.5mm
디텐덤	2.5mm	2.5mm	2.0mm
압력각	20도	20도	20도
전위량	0	0	0
공구 필렛값	1.0mm	1.0mm	1.0mm

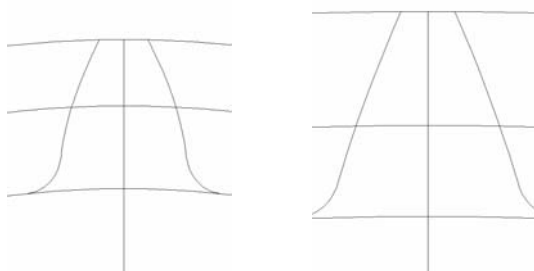
여기서 링기어는 내접기어(Internal Gear) 이므로, 어텐덤과 디텐덤이 일반 평기어와 반대로 되어 있음을 주의하시기 바랍니다.

이상 확정된 3가지의 기어 사양을 모두 GPG에 입력하여 살펴보고, 확인합니다.

7.3. AutoCAD로 작도

GPG로 모두 작도하여 AutoCAD용 스크립트 파일로 저장합니다. ('Save Profile' 단추 사용)

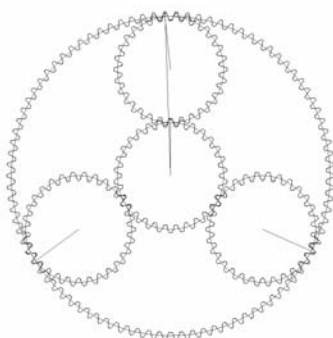
그리고 AutoCAD의 script 명령으로 자동 작도해 보고,



↑ 태양기어, 유성기어

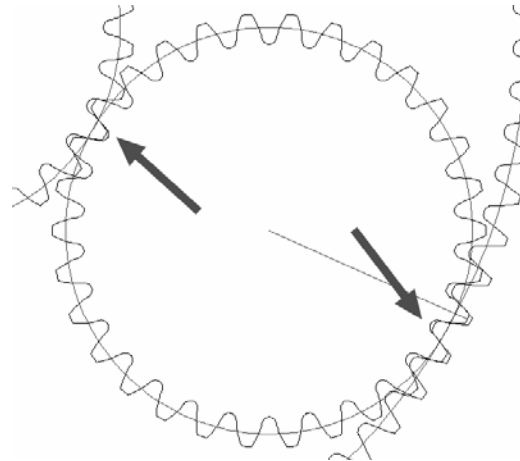
↑ 링기어

편집을 해서 다음과 같이 기어 형상을 배치해 봅니다.



↑ 편집, 배치된 유성기어박스의 레이아웃

그리고 기어 이빨들 사이에 간섭이나 어긋남 같은 것은 없는지, 잘 맞추어 봅시다.



↑ 기어 물림 관계 성립 확인

여기서 그린 기어박스의 경우에는, 유성기어의 3개가 120도 간격으로 배치되어 있으므로 이점에 유의해야 할 것입니다. 링기어와 유성기어 사이에 3의 배수로 나누어 떨어져야만 기어의 물림 관계가 성립이 되니까요.

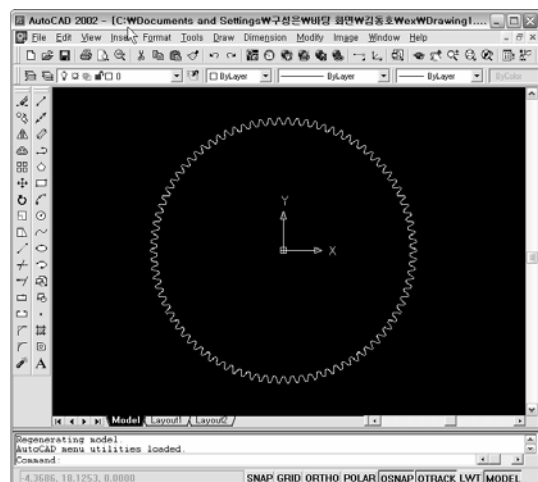
7.5. Pro/Engineer 2001로 불러들이기

여러가지 3D 설계도구가 있겠으나, 기구설계자들이 애용하는 Pro/E에서 이 형상을 불러들여 다루어 보겠습니다.

AutoCAD에서 작도된 형상을 Pro/E에서 읽어들이기 위해서는 일반적인 *.dxf 포맷으로 바꾸어 주어야죠.

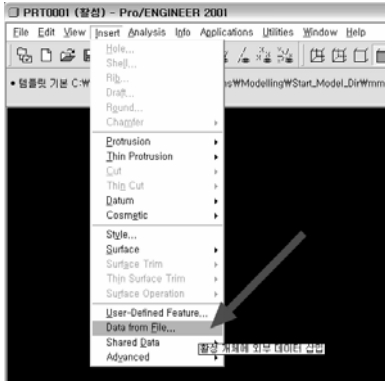
이때 물론 절대좌표계에서 기어의 중심이 원점에 있도록 위치를 조정해 주고, 또한 block 설정을 풀어주는 것(explode)이 필요합니다. (block으로 설정된 요소는 Pro/E에서 읽어들이지 못하더군요.)

그렇게 각각의 기어를 *.dxf로 저장해 둡니다.



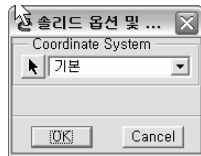
↑ 원점을 중심에 맞추고, *.dxf 파일로 저장하세요.

이제 Pro/E를 시작하고, 새로운 *.prt 파일을 만들어서 엽니다. 'Insert → Data from file...' 메뉴를 선택합니다.



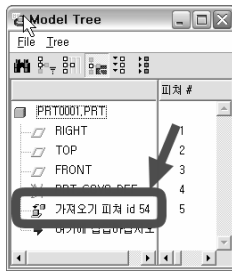
↑ Insert → Data from file...

그러면 파일 선택하는 팝업창이 뜨는데, 아까 저장해 둔 *.dxf 파일을 찾아서 선택합니다. 그러면 *.dxf 파일이 Pro/E의 *.prt 파일 안으로 읽어 들여지는데, 위치를 지정하기 위해 좌표계를 선택하는 팝업이 뜨죠.



↑ 읽어들일때 일치시킬 좌표계 지정

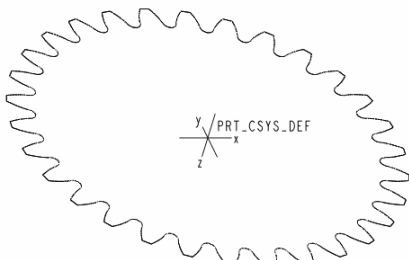
여기서는 그냥 '기본' 상태로 둡시다. OK 하면,



↑ 읽어들인 형상이 feature로 등록됨

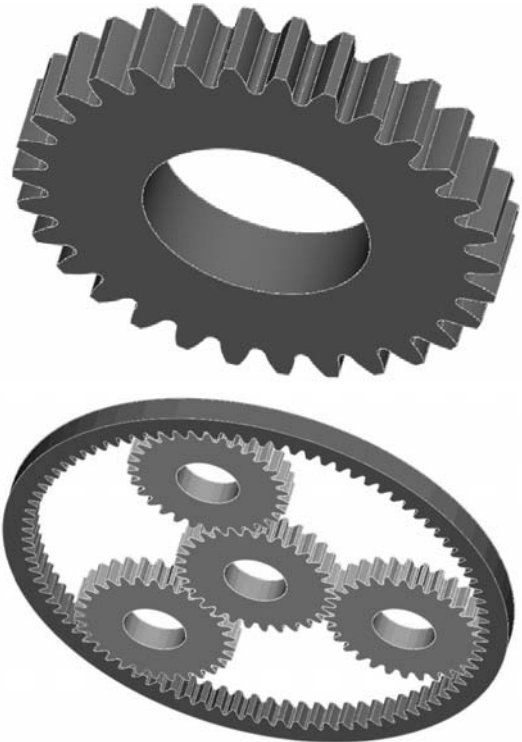
하나의 feature로서 등록이 되어 있을 것입니다.

Pro/E의 배경 화면이 푸른색 등이라면 읽어들여진 모양이 잘 보일 것입니다. 배경 화면이 검은색이라면 안보일 것인데, 이유는 읽어들인 엔티티가 검은색 선으로 나오기 때문에 검은색 배경일때는 안보입니다. 따라서 이때는 선의 색상을 바꾸어 주면 됩니다. (Menu Manager에서 Modify(수정)→Line style(선 유형)→Pick many(복수선택)→Pick box(상자선택)→Pro/E 모델링 화면에서 마우스를 왼쪽 버튼 누른채로 드래그 해서 전체 선택함. 2001버전 기준.)



↑ 읽어들여진 feature의 모습

이제 이것을 이용해서 extrude 명령 등을 이용해서 3D 형상을 만들어 낼 수 있을 것입니다.



↑ 구현된 3D 형상

이 방법은 익숙한 AutoCAD에서 필요한 편집을 완료한 후에 손쉽게 Pro/E 형상으로 간단히 구현 가능하다는 장점이 있습니다.

하지만 잇수가 많아져 엔티티가 증가하면 Pro/E의 스케치 모드(Sketcher Mode)에서 너무 무거워지고, Pro/E 고유의 파라메트릭(Parametric)한 특성을 이용할 수 없게 된다는 단점이 있습니다.

Pro/E에서는 이런 방법 이외에도 외부의 데이터를 받아들이는 다른 방법들도 존재하고 있으므로, 각 사용자의 아이디어에 따라 다양한 방법들이 나올 수 있을 것입니다.

궁극적으로는 Pro/Toolkit 프로그래밍을 통해 Pro/E 전용의 Application을 작성해서 이용하는 것이 가장 좋겠지요.

8. 맺음말

GPG를 이용하여 간단한 기어를 자동 작도하고 설계해 보았습니다. 이보다 한 단계 더 높은 수준의 설계가 필요한 경우에는 상용 기어 설계용 소프트웨어를 이용해 볼 것을 고려할 수 있습니다. 다만 이를 위해서는 기어에 관련된 더 많은 배경지식과 자료가 있어야 그 소프트웨어의 효용을 충분히 발휘 가능하다고 생각합니다.

감사합니다~