마지막 변경일 2018년 5월 7일

** 타원 내부에서의 포락선 **

Geogebra와 수학의 시각화 책의 7.1소절 내용임.

http://min7014.iptime.org/math/2017063002.htm

가장 최근 파일은 링크를 누르면 받아 보실 수 있습니다.

https://goo.gl/HAq93C

http://min7014.iptime.org/math/2018010701.pdf

자료의 수정이 필요한 부분이 있으면 언제든지

민은기 E-mail : min7014@nate.com

이경수 E-mail : ksteach81@gmail.com

으로 연락주시면 감사하겠습니다.

강의록을 보기전에 프로그램 설치를 반드시 읽어보시고 꼭 지오지브라 클래식 5를 설치하시기 바랍니다.

https://goo.gl/wqwJ6v

http://min7014.iptime.org/math/2018011001.pdf

* 주요변경내역 *

2017.06.24 Geogebra와 수학의 시각화 책에 엮어 출간.

차 례

제 1 장

Geogebra를 활용한 연구활동

1.1 타원 내부에서의 포락선

2017년 경남과학고 수학과 선생님들과 학생들에게 수학에서의 연구라는 주제로 강의를 준비하며 그동안 만든 자료를 다시 보는 계기가 되었습니다. 아마도 여러분께 수학에서의 연구에 대한 경험을 들려주는 것이 앞으로의 연구에 도움되리라 생각되어 실제 연구 주제로 생각했던 내용을 재구성하여 강의하겠습니다. 이 연구의 내용은 부산 덕문여자고등학교 정봉균 선생님이 연구한 내용을 듣고 Geogebra로 선생님의 연구에 도움을 드렸던 이야기입니다. 이 강의록을 작성하면서 결국 수학적으로 문제를 해결하게 된 이야기까지 하도록 하겠습니다.

1.1.1 연구하게 된 과정

2016년 2월경 부산 덕문여자고등학교에 근무하시는 정봉균 선생님과 Geogebra에 대하여 이야기를 나눈 일이 있습니다. 선생님께서는 그간 여러 가지 학생지도 경험을 말씀해 주셨습니다. 그 가운데 학생들과 연구활동 했던 경험도 들려주셨습니다. 선생님은 타원내부에서의 빛 반사 실험을 직접 해보셨다고 합니다. 초점에서 빛을 쏘았을 때 빛이 다른 초점으로 가는 것은 잘 알려진 사실입니다. 이런 성질을 직접 구현하기 위해 타원면을 만들고 레이저 포인터를 활용하여 빛 반사 실험을 실제로 해보면서 초점에서 빛을 쏘면 초점으로 가는지를 실험해 보셨다고 합니다. 이 실험을 하는 중에 빛이 타원과 만나면서 계속 반사하게 되는데 빛이 지나는 부분과 지나지 않는 부분의 경계가 어떤 모양이 되는지에 대해 의문을 가지셨다고 합니다. 기하작도 소프트웨어를 활용하여 하나씩 하나씩 여러 번에 걸쳐서 직접 작도를 해보았더니 경계가 재미있는 곡선에 가까이 가는 것을 관찰할 수 있었다고 합니다. 재미있는 것은 초점과 초점사이로 빛을 보내면 빛이 지나는 곳과 지나지 않는 곳의 경계선이 쌍곡선과 비슷한 모양을 이루고 초점과 초점사이 밖으로 빛을 보내면 빛이 지나는 곳과 지나지 않는 곳의 경계가 타원과 비슷한 모양을 이루는 것을 관찰했다고

합니다. 빛이 반사한다는 간단한 아이디어였고 이는 수학적 대상이었습니다. 선생님의 이야기를 듣고 빛이 반사한다는 것은 쉽게 원리를 알 수 있지만 머릿속으로 타원속에서 빛이 지나는 부분과 그렇지 않은 부분의 경계가 무엇이 될지 그려지지 않았습니다. 정봉균 선생님께서는 “Geogebra로 빛 반사를 무한히 반복하는 것을 해 볼 수 있겠는가?” 하고 물으셨고 “시도해 보겠습니다.” 하고 답하였습니다. 선생님과 헤어진 뒤 Geogebra로 수학적 사고실험을 해보기로 마음먹었고 많은 고민 끝에 실험에 성공했습니다.

이 경계선은 포락선에 해당합니다. 포락선의 정의를 위키피디아에서 찾아보니 『포락선(envelope, 包絡線)은 어떤 단일 매개변수에 따라 정의된 무한개의 곡선이 있을 때 그 곡선군의 모든 곡선에 접하는 곡선을 이르는 말이다.』로 되어 있습니다. 이 정의로 봤을 때 경계선의 방정식을 구한다면 이 빛이 지나는 경로는 접선에 해당하는 것이므로 경계선은 빛이 지나는 선에 접하는 포락선으로 볼 수 있는 것입니다.

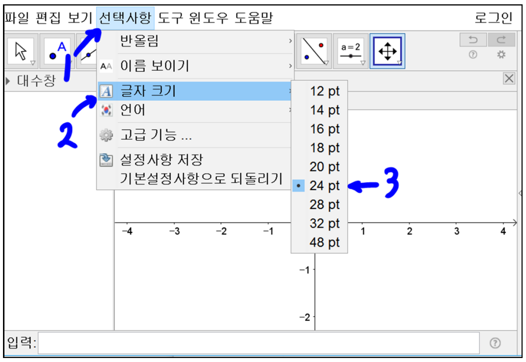
위의 포락선의 정의로 보면 단일 매개변수로 정의되는 곡선군을 만들어야 합니다. 빛 반사로 만들어지는 선들은 어쩌면 이산적인 대상인데 무한히 반사되면서 연속적인 것으로 바뀐 것입니다. 매개변수로 정의되는 연속적인 곡선군을 찾는 것이 어렵고 포락선의 방정식을 찾는 것이 쉽지는 않을 것이라고 보여집니다. 즉 실험을 통해서가 아니면 이 포락선을 쉽게 볼 수 없을 것 같습니다. 수학에서 실험이란 것을 할 수 있구나! 수학에서 연구를 보는 시각이 많이 바뀌게 되었습니다. 순수수학을 많은 시간 공부한 경험 때문인지 수학에서는 보통 이데아적인 것에 가치를 두어 생각만으로 연구 활동을 할 것이라고만 생각했습니다. 이 타원 안에서의 실험을 통해 수학에서는 이데아적인 생각이 컴퓨터를 활용하여 Geogebra라는 소프트웨어를 통해 수학에서 실험이라는 것을 가능하게 한 것입니다.

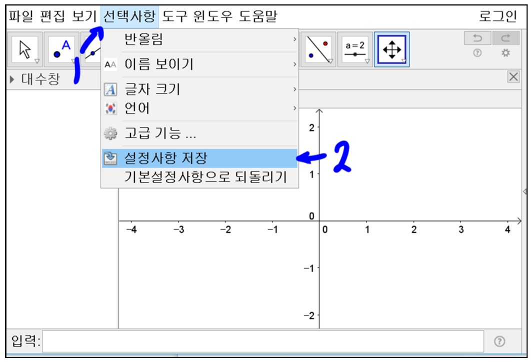
여기까지의 내용은 2017학년도 경남과학고등학교 수학과 현장연구 특강 강의

록에 작성했던 내용입니다. 이후 포락선이 주어진 타원과 같은 초점의 타원과 쌍곡선이란 것을 예측하고 Geogebra로 확인했습니다. 이제 남은 것은 수학적 증명입니다. 증명도 Geogebra의 동적기하를 활용하여 관찰하며 해냈습니다.

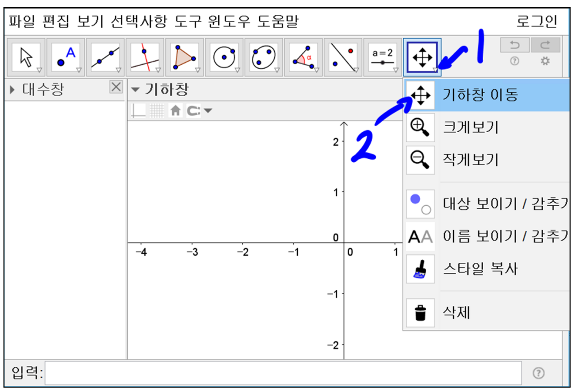
Geogebra로 경계선에 대한 작도 방법만을 소개하면 되겠지만 이러한 아이디어를 생각하게 된 계기가 어쩌면 더 중요하다고 생각되어 정봉균 선생님과의 에피소드를 적게 되었습니다. 다들 알고 있는 것을 직접 실험을 해보신 것입니다. 그러다가 새로운 아이디어를 얻어 발전시킨 것입니다. 그동안 Geogebra를 많이 사용했지만 수학에서 실험을 할 수 있다는 경험을 직접 얻은 것은 처음입니다. 지금부터 Geogebra를 활용하여 위의 아이디어를 직접 실험을 통해 해보도록 하겠습니다. 수학적 증명까지 보여드리도록 하겠습니다.

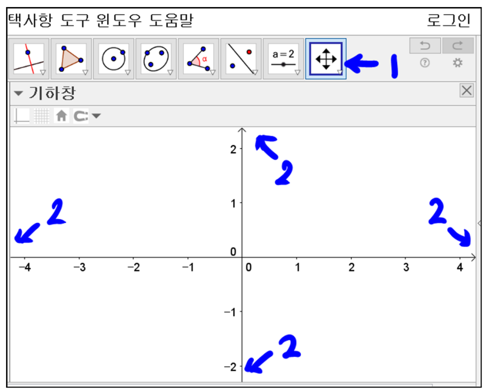
1.1.2 Geogebra 기본 도구로 타원 내부에서의 빛 반사



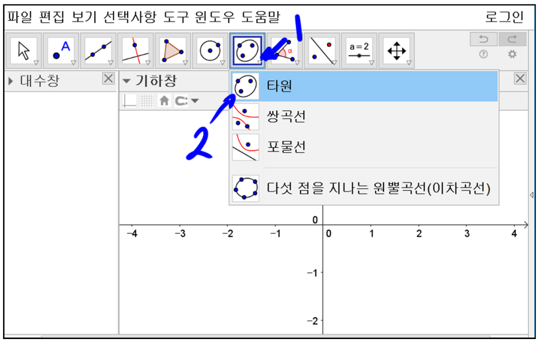


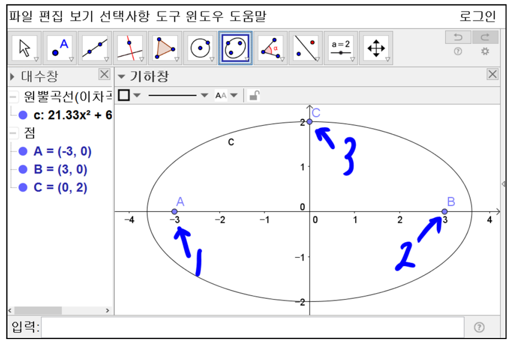
가독성이 좋도록 선택사항을 선택하고 글자 크기를 선택하고 24pt로 선택합니다. 다음에 실행할 때 설정이 유지되도록 선택사항을 선택하고 설정사항 저장을 선택합니다.



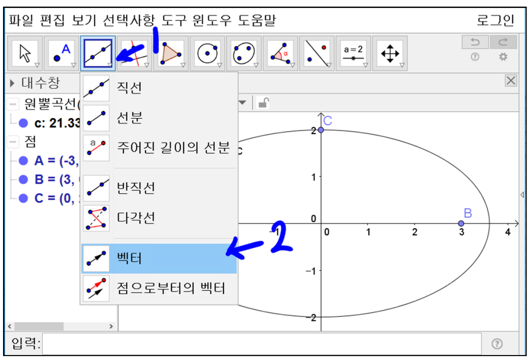


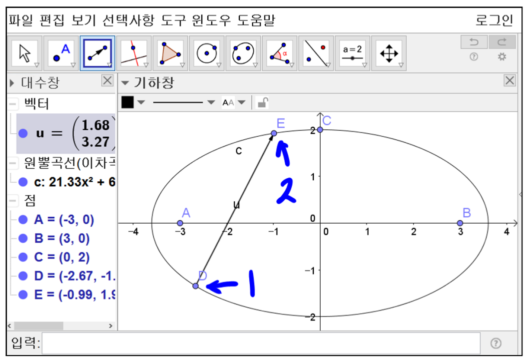
'기하창 이동' 도구를 선택하고 기하창의 직교좌표의 원점을 가운데로 위치시키고 '크게보기' 또는 '작게보기' 도구를 사용하여 기하창을 확대하거나 축소해서 x축의 범위가 -4.3에서 4.3이 되도록 하고 y축의 범위가 -2.3에서 2.3이 되도록 합니다.



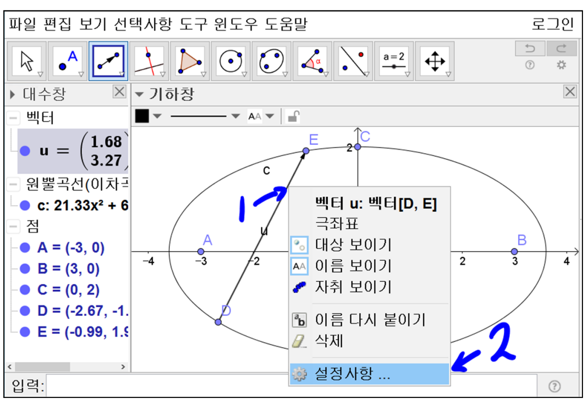


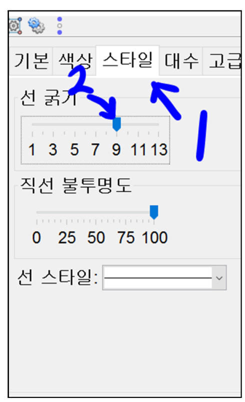
‘타원’ 도구를 선택합니다. $(-3,0)$을 선택하고 $(3,0)$을 선택하고 $(0,2)$를 선택하면 타원이 그려집니다.



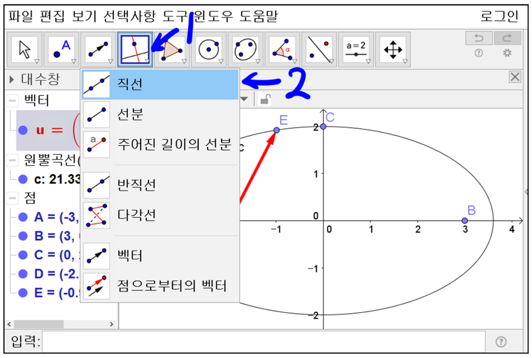


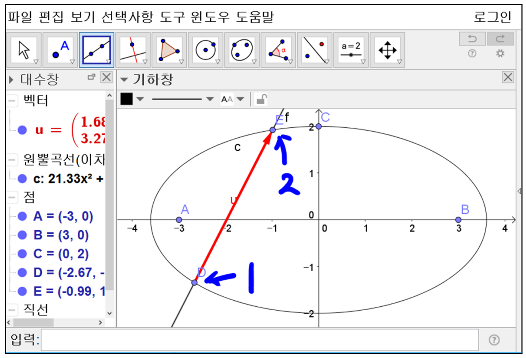
‘벡터’ 도구를 선택합니다. 3사분면 쪽의 타원 위에 점을 찍고 2사분면 쪽의 타원 위에 점을 찍으면 벡터가 만들어 집니다.



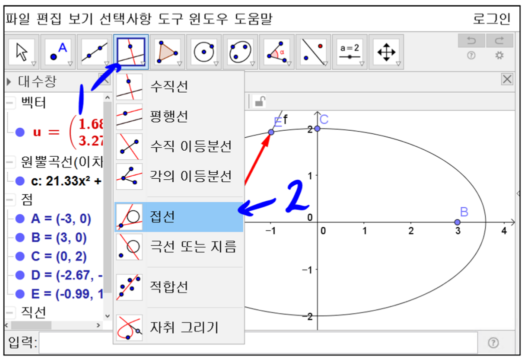


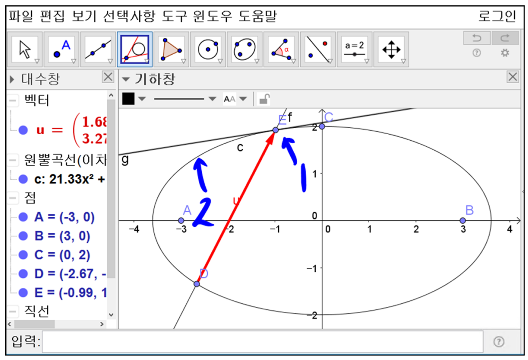
벡터 u위에 마우스를 놓고 오른쪽 버튼을 클릭하고 설정사항을 선택합니다. 색상을 누르고 빨간색을 선택합니다. 스타일을 누르고 선 굵기를 9로 합니다.



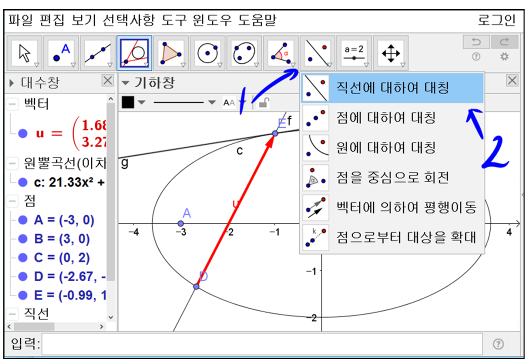


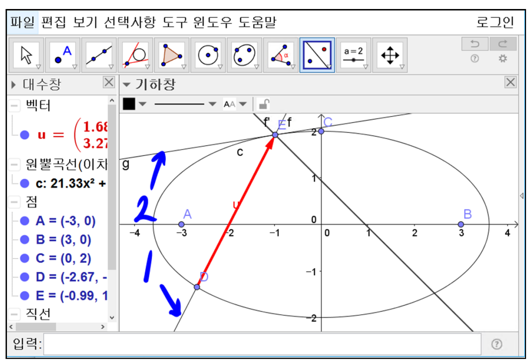
'직선' 도구를 선택합니다. 차례로 점 D와 점 E를 선택하여 직선을 그립니다.



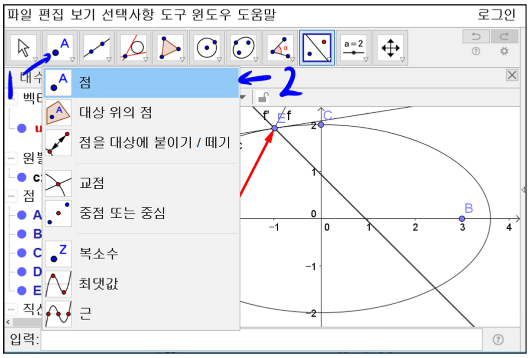


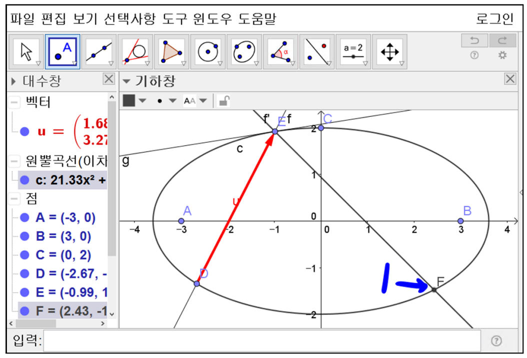
'접선' 도구를 선택합니다. 차례로 점 E와 타원 c를 선택하여 타원 c 위의 점 E에 접하는 접선을 그립니다.



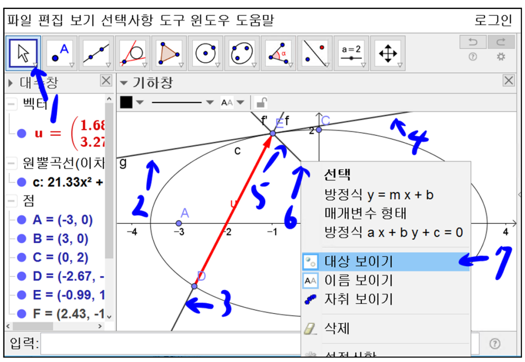


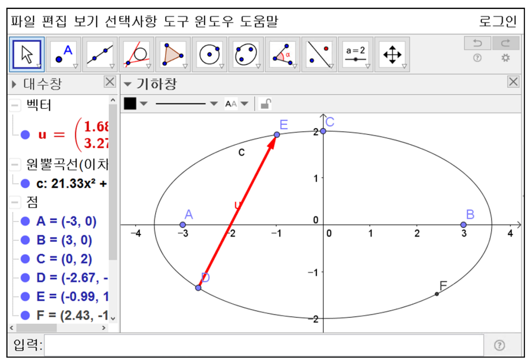
'직선에 대하여 대칭' 도구를 선택합니다. 차례로 직선 f와 직선 g를 선택하여 직선 f를 직선 g에 대칭시킨 직선 f′를 그립니다.



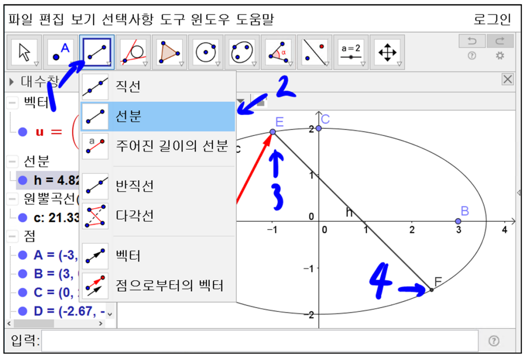


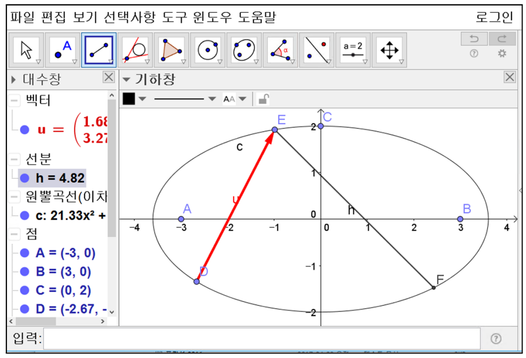
'점' 도구를 선택합니다. 마우스를 직선 f′과 타원 c의 교점에 가져가면 직선 f′과 타원 c가 동시에 진하게 강조됩니다. 이때 마우스를 클릭하면 직선 f′과 타원 c의 교점 F가 그려집니다.





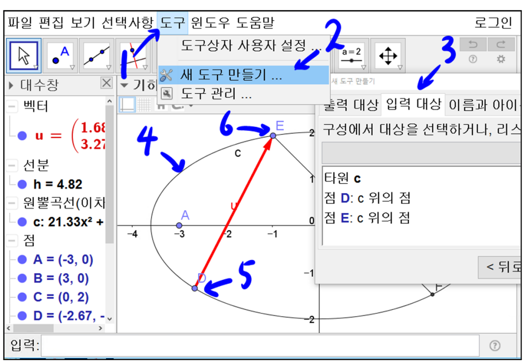
'이동' 도구를 선택합니다. [Ctrl]를 누른 상태에서 선 g, 선 f, 선 f′을 차례로 선택합니다. 선 g, 선 f, 선 f′이 동시에 진하게 강조되면서 선택이 됩니다. 이때 선 f′ 위에 마우스를 놓고 오른쪽 버튼을 클릭하고 대상 보이기를 선택하면 선 g, 선 f, 선 f′이 화면에서 사라집니다. 절대로 삭제를 선택하지 마시기 바랍니다. 만약 삭제를 선택하면 이 선들에 의존한 점 F도 삭제됩니다.

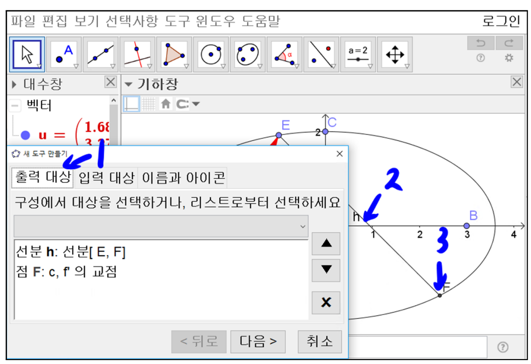




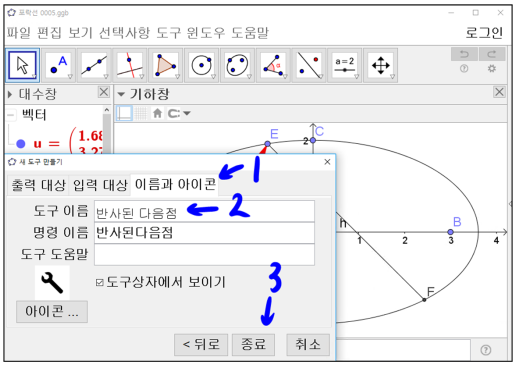
'선분' 도구를 선택합니다. 점 E와 점 F를 차례로 선택합니다. 선분 h가 그려집니다.

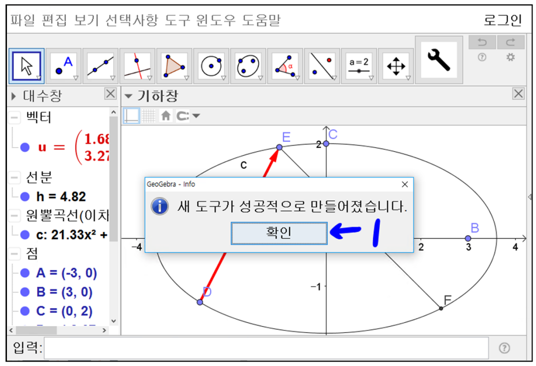
지금까지 한 것은 타원내부에서 빛이 점 D에서 벡터 u 방향으로 빛을 쏘았을 때 점 E에서 반사되어 점 F에 빛이 도달한 것을 나타내 보았습니다. 위와 같은 방법으로 이번에는 점 E를 시작점으로 하고 점 F를 도달점으로 보고 빛의 다음 도달점을 찾을 수 있습니다. 하지만 너무 지루하고 반복적인 작업이 될 것입니다. 다행이도 Geogebra에는 이런 반복적인 작업을 한 번에 할 수 있는 방법이 있습니다. 지금까지 몇 가지 도구를 사용하였습니다. 이와 같은 도구를 만들 수 있는 기능이 있습니다. 점 D와 점 E를 차례로 선택하면 자동으로 점 F와 선분 EF가 그려지도록 할 수 있습니다.



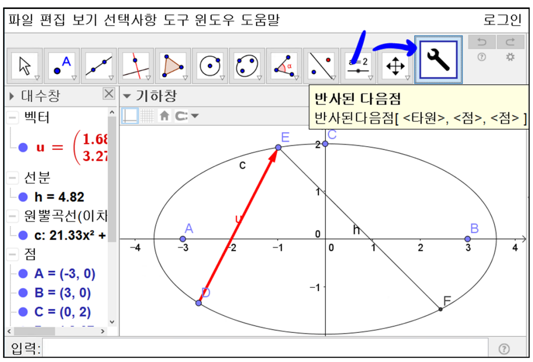


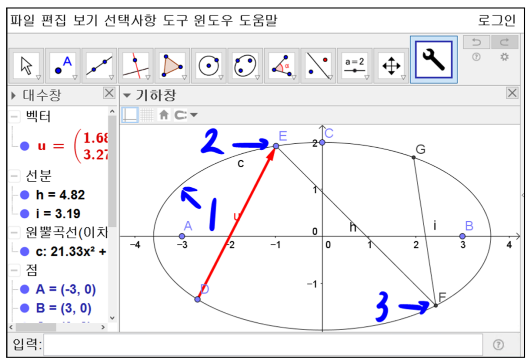
맨 상단에 있는 파일, 편집, 보기, 선택사항, 도구, 윈도우, 도움말 중 도구를 선택하고 새 도구 만들기를 선택합니다. 새 도구 만들기 창에 입력대상을 선택합니다. 기하창에 타원 c, 점 D, 점 E를 차례로 선택합니다. 출력 대상을 선택합니다. 선분 h와 점 F를 선택합니다.





이름과 아이콘을 선택합니다. 도구 이름에 ‘반사된 다음점’이라고 입력합니다. 종료를 누릅니다. 확인을 누릅니다. 도구가 만들어 졌습니다. 도구들이 위치해 있는 곳에 지금 만든 ‘반사된 다음점’ 도구가 있는 것을 확인할 수 있습니다.

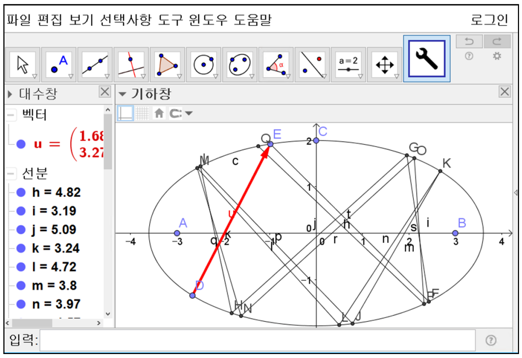


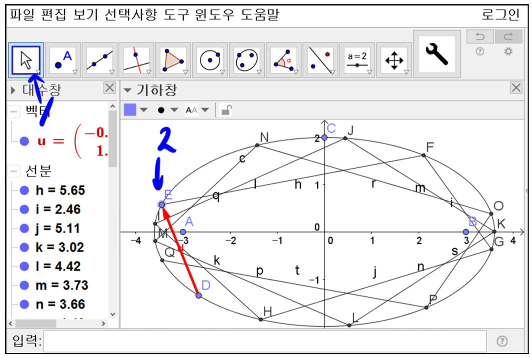


‘반사된 다음점’ 도구를 선택합니다. 친절하게도

$$\text{반사된다음점}[<\text{타원}>,<\text{점}>,<\text{점}>]$$

이 보입니다. 타원 c, 점 E, 점 F를 차례로 선택합니다. 선분 i와 점 G가 자동으로 그려집니다.

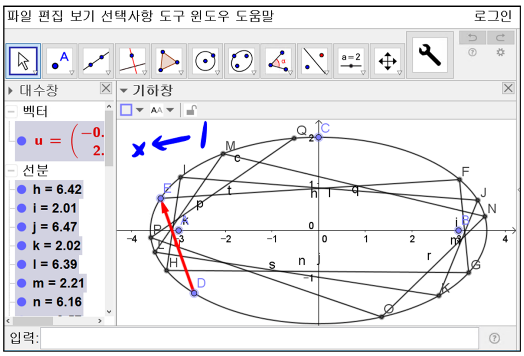


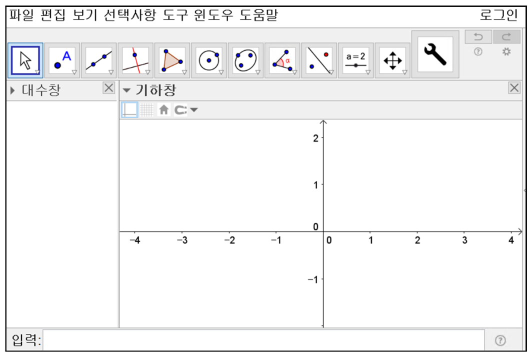


이와 같은 방법으로 ‘반사된 다음점’ 도구를 사용하여 10개 정도 반사된 점들을 차례로 더 그립니다. 도구를 만들지 않았다면 지루한 작업이 되었을 것입니다. 빛이 지나는 곳과 빛이 지나지 않는 경계가 마치 쌍곡선의 일부가 되는 것으로 보여집니다. ‘이동’ 도구를 선택합니다. 점 E를 초점 A와 초점 B 사이 밖으로 빛이 지나가도록 위치시켜 봅니다. 빛이 지나는 곳과 빛이 지나지 않는 경계가 마치 타원인 것처럼 보입니다. 이처럼 Geogebra 기본도구를 사용하고 새로운 도구를 만들어서 간단하게 수학에서 사고실험을 직접 Geogebra로 해볼 수 있습니다.

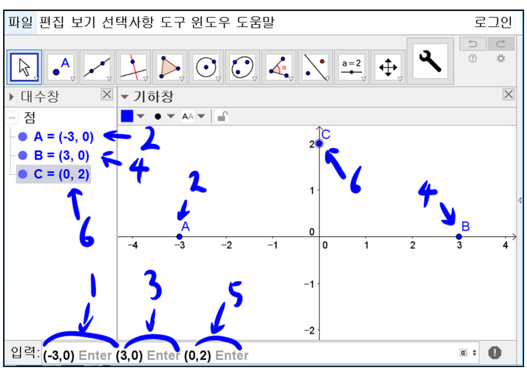
1.1.3 Geogebra 스크립트를 활용한 프로그래밍으로 타원 내부에서의 빛 반사

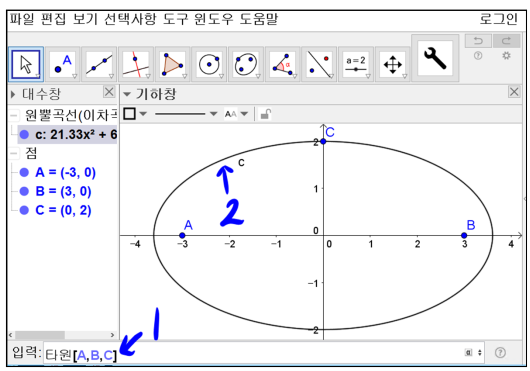
위의 결과물이 마음에 들지 않는다면 반사된 점들을 더 많이 그려보면 더 명확하게 볼 수 있을 것입니다. 100개 정도만 도구를 사용하더라도 많은 시간이 걸립니다. 컴퓨터를 활용하는 이유가 반복적인 작업의 자동화일 것입니다. 이제부터는 Geogebra 스크립트를 활용하여 약간의 프로그래밍을 해보도록 하겠습니다.





기하창 빈곳에 마우스를 위치시키고 선택합니다. [Ctrl]를 누른 상태에서 [A]를 누릅니다. 모든 개체들이 선택될 것입니다. [Delete]를 누릅니다. 그동안 했던 작업이 사라졌습니다.

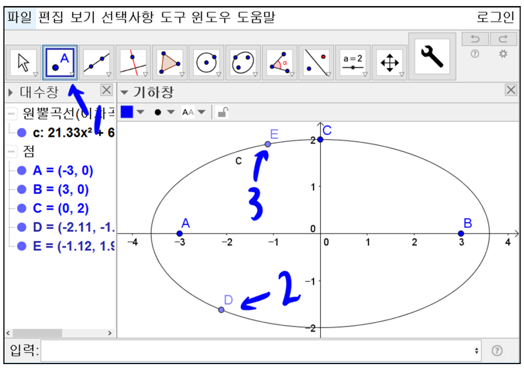


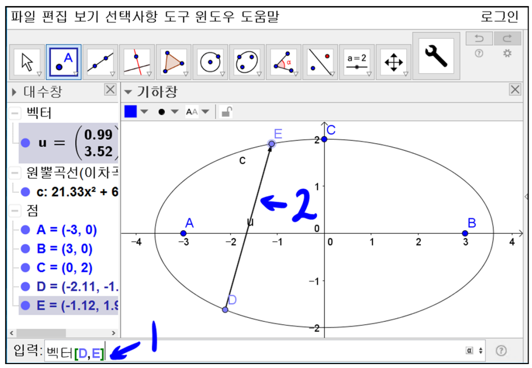


쓰일 명령어는 다음과 같습니다.

(a, b) (a, b는 실수)	x좌표가 a이고 y좌표가 b인 점
타원[A, B, C] (A, B, C는 점)	점 A와 점 B를 초점으로 하고 점 C를 지나는 타원

이번에는 명령어 입력으로 타원을 만들어 보겠습니다. 입력에 $(-3, 0)$을 넣고 [Enter] 를 누릅니다. 입력에 $(3, 0)$을 넣고 [Enter] 를 누릅니다. 입력에 $(0, 2)$을 넣고 [Enter] 를 누릅니다. 점의 좌표를 입력하고 [Enter] 를 누르면 점이 만들어 집니다. 타원[A, B, C]를 입력에 넣고 [Enter] 를 누릅니다. 초점을 점 A와 점 B로 하고 점 C를 지나는 타원 c가 만들어 졌습니다.

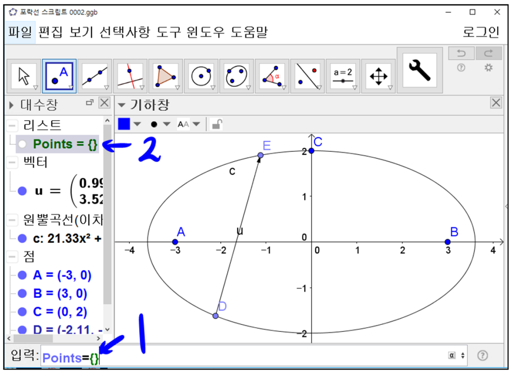


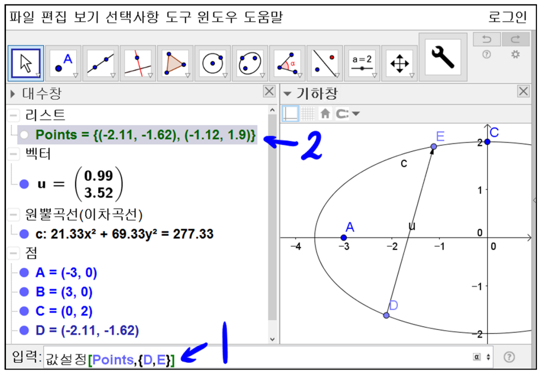


쓰일 명령어는 다음과 같습니다.

벡터[A, B] (A, B는 점)	시점이 점 A이고 종점이 점 B인 벡터

'점' 도구를 선택하고 3사분면에 위치한 타원 위에 점을 하나 찍고 2사분면에에 위치한 타원 위에 점을 찍습니다. 입력창에 벡터[D, E] 를 입력하고 [Enter] 를 누릅니다. 점 D를 출발점으로 하고 점 E를 끝점으로 하는 벡터 u가 그려졌습니다.

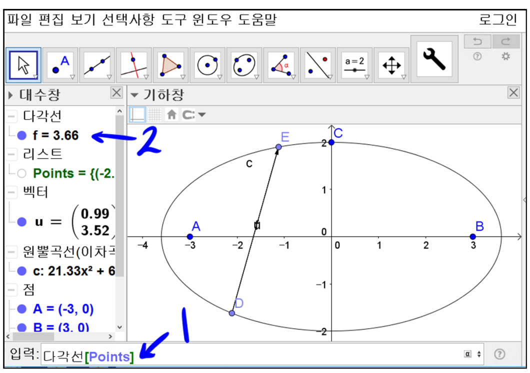


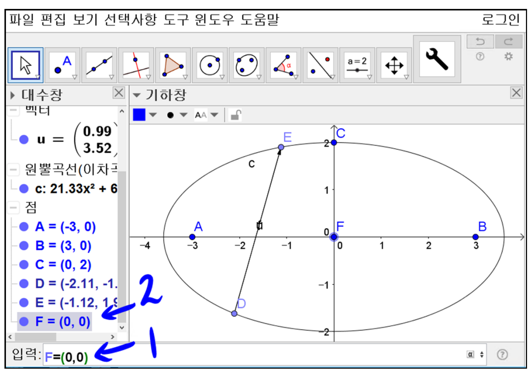


쓰일 명령어는 다음과 같습니다.

{}	빈 리스트
$\{\mathrm{A}_1, \mathrm{A}_2, \ldots, \mathrm{A}_\mathrm{n}\}$ $(\mathrm{A}_1, \mathrm{A}_2, \ldots, \mathrm{A}_\mathrm{n}$는 개체)	$\mathrm{A}_1, \mathrm{A}_2, \ldots, \mathrm{A}_\mathrm{n}$ 순서의 리스트
값설정[A, B] (A, B는 개체)	A를 B로 만듦

입력창에 Points = {}를 입력합니다. 리스트에 Points = {}라는 리스트가 생성됩니다. 리스트는 집합과 비슷하지만, 원소의 내용을 넣은 순서대로 위치하고 중복이 가능합니다. 입력창에 값설정[Points, {D, E}]를 입력합니다. 리스트 Points의 내용은 $\{(-2.11, -1.62), (-1.12, 1.9)\}$라고 변경됩니다.

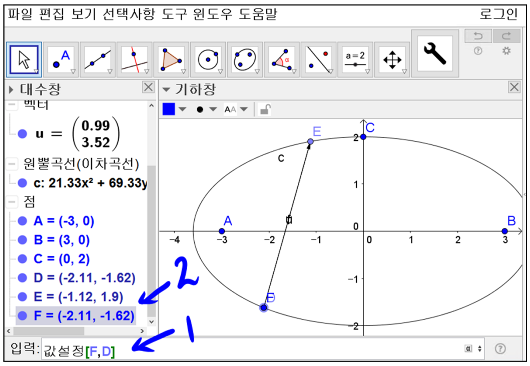


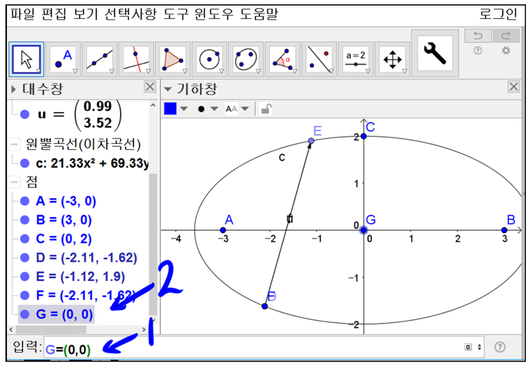


쓰일 명령어는 다음과 같습니다.

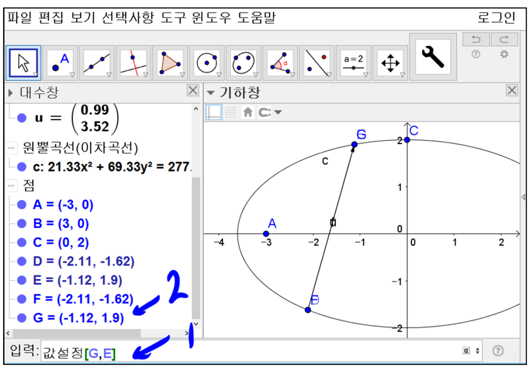
다각선[List] (List는 점을 원소로 하는 리스트)	List의 점으로 구성된 다각선

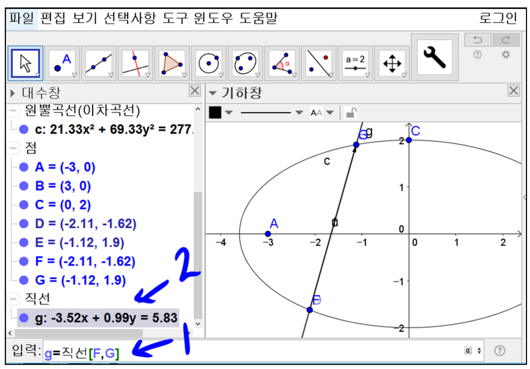
입력창에 다각선[Points]를 입력합니다. 대수창에 다각선 f가 생성됩니다. 입력창에 F = (0, 0)을 입력합니다. 대수창에 점 F가 생성됩니다.





입력창에 값설정[F, D]를 입력합니다. 점 F의 위치가 점 D의 위치로 바뀝니다. 입력창에 G = (0, 0)을 입력합니다. 대수창에 점 G가 생성됩니다.

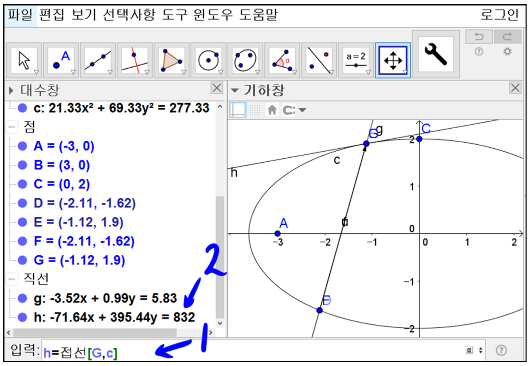


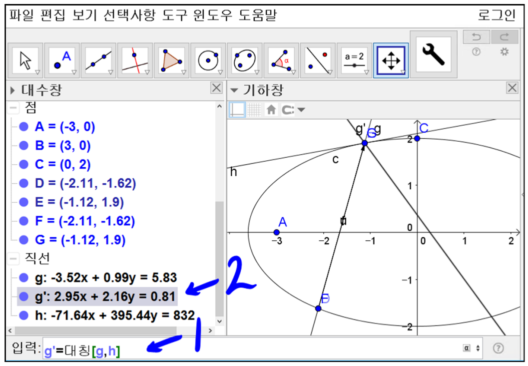


쓰일 명령어는 다음과 같습니다.

직선[A, B] (A, B는 점)	점 A와 점 B를 지나는 직선

입력창에 값설정[G, E]를 입력합니다. 점 G의 위치가 점 E의 위치로 바뀝니다. 입력창에 g = 직선[F, G]를 입력합니다. 대수창에 점 F, 점 G를 지나는 직선 g가 생성됩니다.

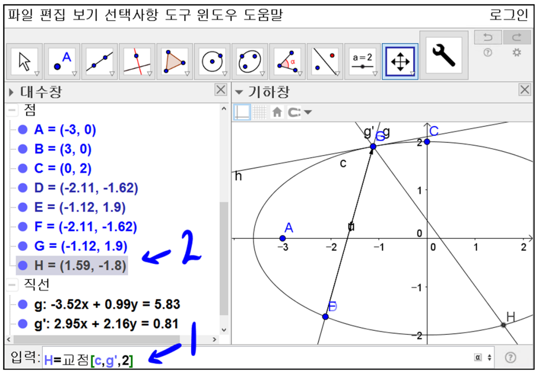


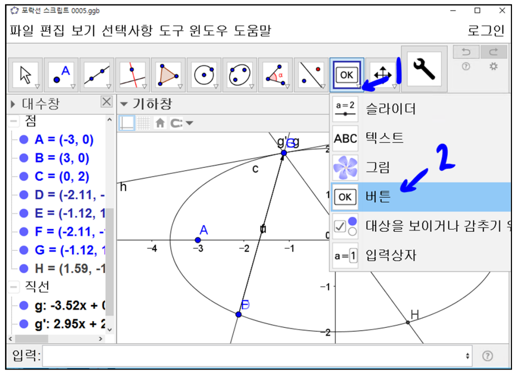


쓰일 명령어는 다음과 같습니다.

접선[A, c] (A는 점, c는 곡선)	– 점 A가 곡선 c 위에 있을 때 : 점 A에서 곡선 c에 접하는 접선 – 점 A가 곡선 c 위에 있을 때 : 점 A에서 곡선 c에 접하는 접선들의 리스트
대칭[A, g] (A는 개체, a는 직선)	개체 A가 직선 g에 대칭인 개체

입력창에 h = 접선[G, c]를 입력합니다. 대수창에 점 G에서 타원 c에 접하는 접선 h가 생성됩니다. 입력창에 g′ = 대칭[g, h]를 입력합니다. 대수창에 직선 g를 직선 h에 대칭인 직선 g′가 생성됩니다.

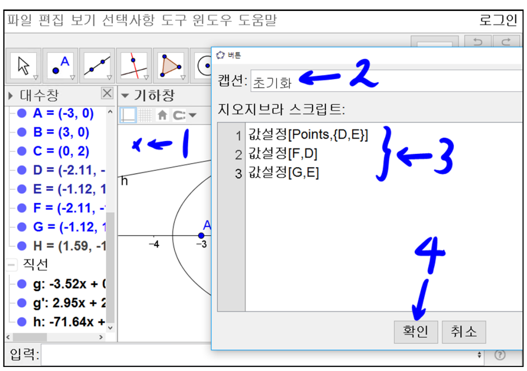


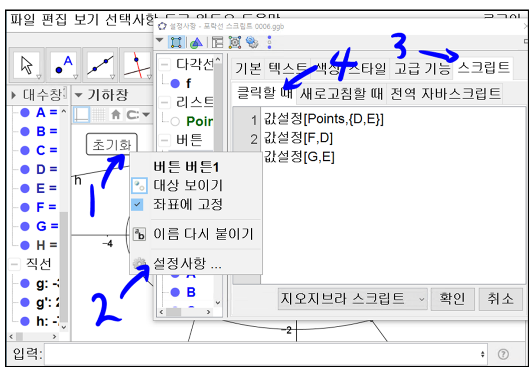


쓰일 명령어는 다음과 같습니다.

교점[g, c, n] (g는 선, c는 곡선, n는 자연수)	g와 c의 n번째 교점

입력창에 H = 교점[c, g′, 2]를 입력합니다. 타원 c와 직선 g′의 교점들 중 2번째 점 H가 생성 됩니다. '버튼' 도구를 선택합니다. 첫 번째 교점과 두 번째 교점을 말하려면 직선에 방향성이 있어야 합니다. 직선[A, B]와 직선[B, A]로 같은 직선을 정의하더라도 겉모습은 똑 같지만 내부적으로 처리될 때 A에서 B라는 방향과 B에서 A 라는 방향을 포함하고 있다고 보면 이해할 수 있을 것 같습니다.

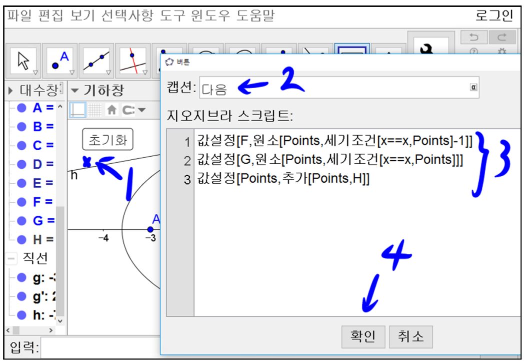


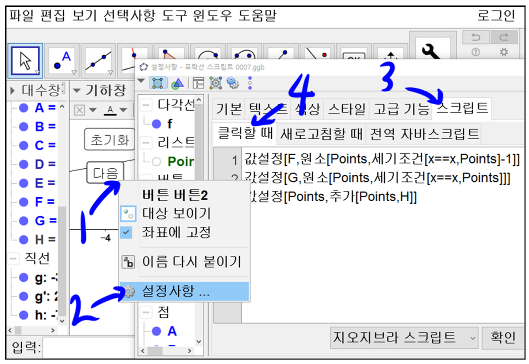


'x' 표시한 곳에 마우스를 위치시키고 선택합니다. 버튼창의 캡션에 초기화라 입력합니다. Geogebra 스크립트에

1 번째 줄 : 값설정[Points, {D, E}]
2 번째 줄 : 값설정[F, D]
3 번째 줄 : 값설정[G, E]

을 1번째 줄에서 3번째 줄까지 입력합니다. 확인을 선택합니다. 기하창에 초기화 버튼이 생깁니다. 초기화 버튼에 마우스를 위치시키고 오른쪽 버튼을 클릭합니다. 설정사항을 선택합니다. 설정사항 창에서 스크립트 탭에 '클릭할 때' 탭을 선택하면 입력한 스크립트를 확인할 수 있습니다.





쓰일 명령어는 다음과 같습니다.

원소[List, n] (List는 리스트, n는 자연수)	List의 n번째 원소
세기조건[조건(x), List] (조건(x)는 x의 조건부 명제 , List는 리스트)	List의 원소 중 조건(x)를 만족하는 만족하는 원소의 개수
추가[List, A] (List는 리스트, A는 개체)	List에 A를 마지막에 추가한 리스트

'x' 표시한 곳에 마우스를 위치시키고 선택합니다. 버튼창의 캡션에 '다음' 이라 입력합니다. Geogebra 스크립트에

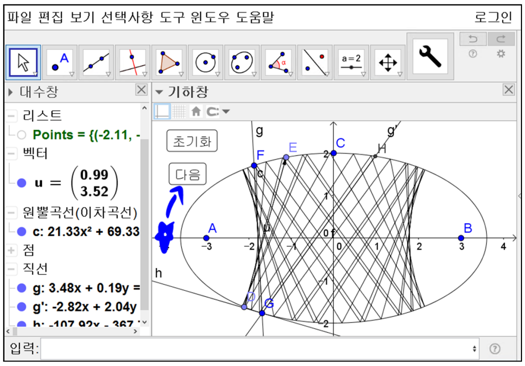
1 번째 줄 : 값설정[F, 원소[Points, 세기조건[x == x, Points] − 1]]
2 번째 줄 : 값설정[G, 원소[Points, 세기조건[x == x, Points]]]
3 번째 줄 : 값설정[Points, 추가[Points, H]]

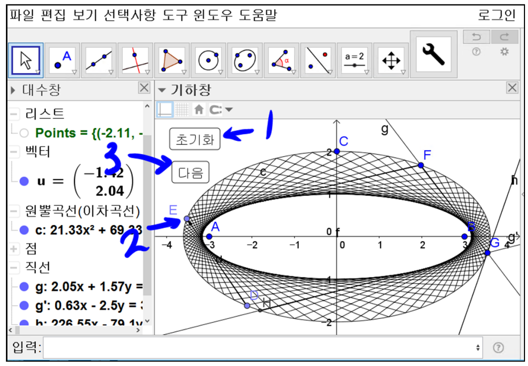
을 1 번째 줄에서 3번째 줄까지 입력합니다. 확인을 선택합니다. 기하창에 다음 버튼이 생깁니다. 다음 버튼에 마우스를 위치시키고 오른쪽 버튼을 클릭합니다. 설정사항을 선택합니다. 설정사항 창에서 스크립트 탭에 '클릭할 때' 탭을 선택하면 입력한 스크립트를 확인할 수 있습니다. 각 줄의 의미를 살펴보겠습니다.

1 번째 줄 : 값설정[F, 원소[Points, 세기조건[x == x, Points] − 1]]
세기조건 명령어로 리스트 Points의 원소 중 x == x 조건을 만족하는 개수를 구합니다. 자기 자신과 같은 것의 개수이므로 리스트 Points의 원소의 개수에 해당합니다. Points의 마지막에서 두 번째 원소인 점의 위치를 점 F에 위치시킨다는 의미입니다.

2 번째 줄 : 값설정[G, 원소[Points, 세기조건[x == x, Points]]]
Points의 마지막 원소인 점의 위치를 점에 위치시킨다는 의미입니다.

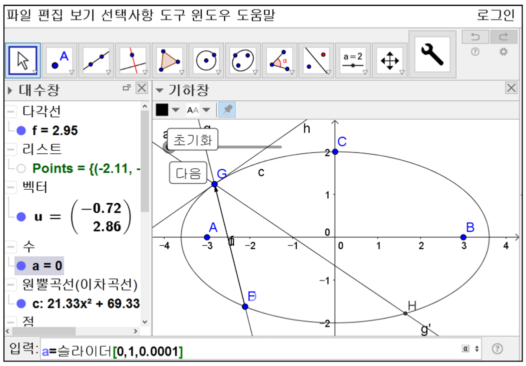
3 번째 줄 : 값설정[Points, 추가[Points, H]]
추가 명령어를(명령어 명칭이 추가임) 사용하여 리스트 Points의 마지막에 점 H의 위치를 추가한 리스트를 다시 리스트 Points로 리스트의 내용을 바꾸란 의미입니다.

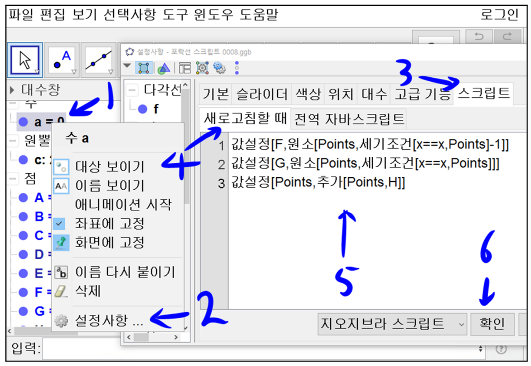




위에서 만든 다음 버튼을 눌러보시기 바랍니다. 한 번 눌렀을 때는 똑 같아 보일 것이지만 두 번째 눌렀을 때는 직선 g, 직선 h, 직선 g′, 점 H, 다각선 f가 바뀌는 것을 볼 수 있습니다. 계속 눌러보시기 바랍니다. 새 도구를 만들었을 때처럼 반사된 다음 점과 선분이 그려집니다. 편리하게도 점이나 타원을 선택하지 않아도 다음 점과 반사된 선이 그려집니다. 다음 버튼을 백번 또는 천번을 누르면 백번 또는 천번 실행해서 얻은 반사된 결과를 볼 수 있습니다. 앞에서 힘들여서 했었는데 너무 쉽습니다. 이렇게 결과가 나온 것은 직선 g, 직선 h, 직선 g′, 점 H 을 만들 때 점 F와 점 G에 의해서 결정되어지기 때문입니다. 점 F와 점 G를 리스트 Points의 마지막 두 점의 위치로 바꾸어 주기만 하면 반사된 마지막 점 H가 결정되고 이 점 H의 위치를 리스트 Points의 마지막 위치에 추가해주면 다각선이 추가된 점까지 그려지는 것입니다. 이번에는 초점 밖으로 빛이 지날 경우의 빛이 지나는 위치를 보겠습니다. 우선 초기화 버튼을 누릅니다. 조금 전 다음 버튼으로 그려졌던 선들이 없어졌습니다. 초기화 버튼의 스크립트 내용을 다시 보면 왜 그렇게 되는지는 알 수 있습니다. 이제 점 E를 움직여 벡터 u가 두 초점에 해당한 점 A와 점 B 밖으로 지나게 위치하게 합니다. 다음 버튼을 적당히 눌러서 빛의 경로를 관찰합니다. 빛이 초점 사이로 가는 경우와 초점 밖으로 가는 경우 각각은 빛이 지나는 부분과 지나지 않은 부분이 다음을 누른 횟수가 많을수록 어떤 포락선이 됨을 좀 더 명확히 보여집니다.

지금까지 한 것만으로도 수학적 사고실험을 Geogebra로 잘 수행했고 충분해 보입니다. 우리는 자동화할 수 있는 컴퓨터를 사용하고 있습니다. 지금까지 버튼에 스크립트를 입력하여 간단한 프로그램을 작성해서 자동화 되었지만 다음 버튼을 백번까지는 눌러볼 수 있겠지만 천번, 만번은 생각해봐야 할 것 같습니다. 이제부터는 버튼 누르지 않고 자동으로 그려지도록 해 보겠습니다.





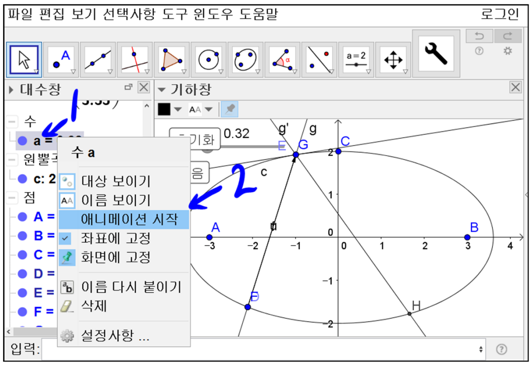
쓰일 명령어는 다음과 같습니다.

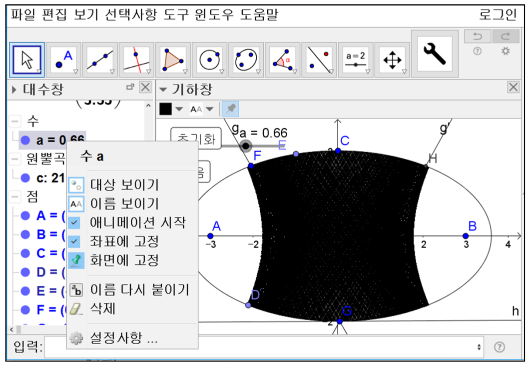
슬라이더$[a, b, c]$ ($a < b, c > 0$)	구간 $[a, b]$에 증가분 c를 가지는 슬라이더

입력창에 $a =$ 슬라이더$[0, 1, 0.0001]$을 입력합니다. 대수창에 수 a가 만들어 집니다. 대수창의 수 a에 마우스를 위치해 놓고 마우스 오른쪽 버튼을 클릭합니다. 설정사항을 선택합니다 스크립트 탭에 '새로고침할 때' 탭을 선택합니다.

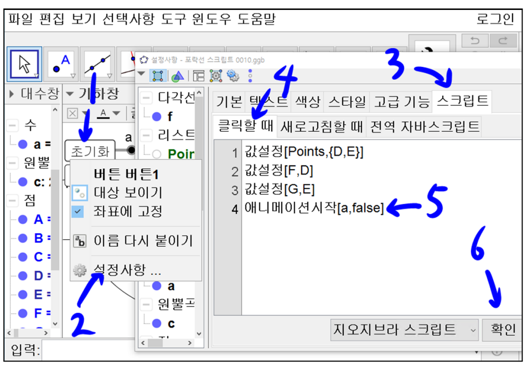
1 번째 줄 : 값설정[F, 원소[Points, 세기조건[$x == x$, Points] $- 1$]]
2 번째 줄 : 값설정[G, 원소[Points, 세기조건[$x == x$, Points]]]
3 번째 줄 : 값설정[Points, 추가[Points, H]]

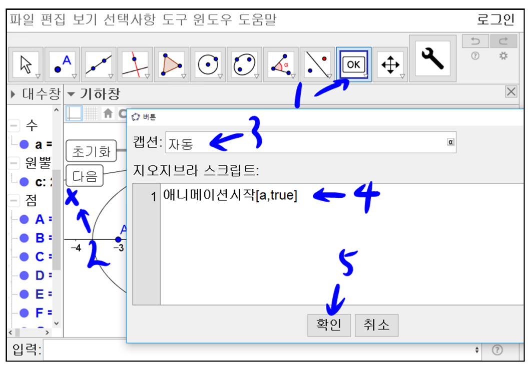
을 직접 입력하거나 다음 버튼의 설정사항에 있는 스크립트 탭에 '클릭할 때' 탭의 내용을 복사해서 입력합니다. 확인을 선택합니다. 수 a의 값이 변하면 새로고침할 때에 입력한 스크립트가 실행이 되도록 한 것입니다.





대수창의 수 a에 마우스를 위치해 놓고 오른쪽 버튼을 클릭합니다. 애니메이션 시작을 선택합니다. 수 a의 값이 변하면서 자동으로 빛의 경로가 그려지는 것이 보여집니다. 어느 정도 지나면 빛이 지나는 곳이 색칠을 한 것처럼 채워집니다. 이제 다시 대수창의 수 a에 마우스를 위치해 놓고 오른쪽 버튼을 클릭합니다. 현재 상태는 애니메이션 시작이 선택되어 체크된 상태입니다. 다시 애니메이션 시작을 선택하면 체크가 해제되면서 수 a가 변하지 않게 됩니다. 이것으로 충분하다고 생각할 수 있지만 아직 마지막 단계가 남았습니다. 수 a 값이 변하게 하는 자동 버튼과 멈춤 버튼을 만들고 초기화 스크립트에 수 a의 값이 변하지 않도록 하는 명령어를 추가하도록 하겠습니다.





쓰일 명령어는 다음과 같습니다.

애니메이션시작[A, true] (A는 개체)	A개체의 애니메시션을 시작
애니메이션시작[A, false] (A는 개체)	A개체의 애니메시션을 멈춤

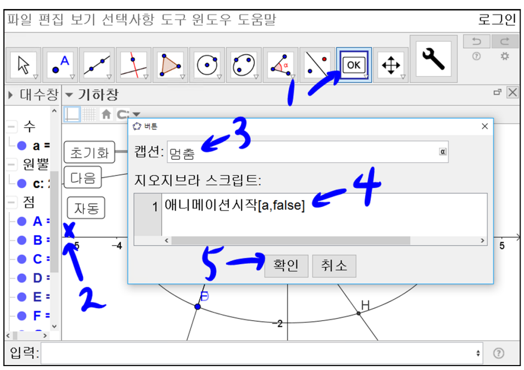
초기화 버튼에 마우스를 위치시키고 마우스 오른쪽 버튼을 클릭합니다. 설정사항을 선택합니다. 스크립트 탭에 '클릭할 때' 탭을 선택합니다. 스크립트에

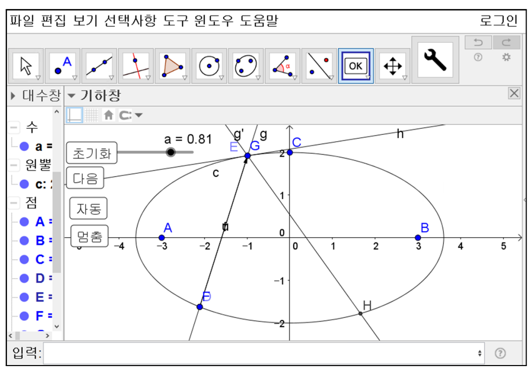
1 번째 줄 : 값설정[Points, {D, E}]
2 번째 줄 : 값설정[F, D]
3 번째 줄 : 값설정[G, E]
4 번째 줄 : 애니메이션시작[a, false]

와 같이 4 번째 줄을 추가로 입력합니다. 확인을 선택합니다. '버튼' 도구를 선택합니다. 마우스를 'x' 표시한 곳을 선택합니다. 캡션에 자동이라고 적고 Geogebra 스크립트에

1 번째 줄 : 애니메이션시작[a, true]

라고 입력합니다. 확인을 선택합니다.



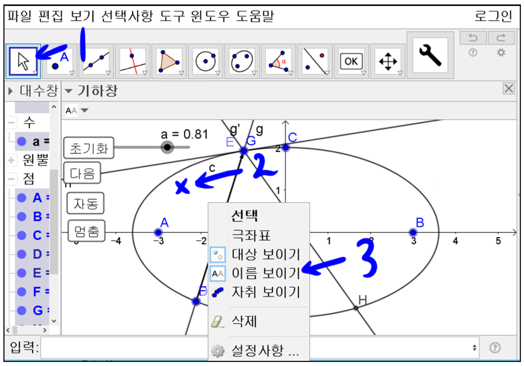


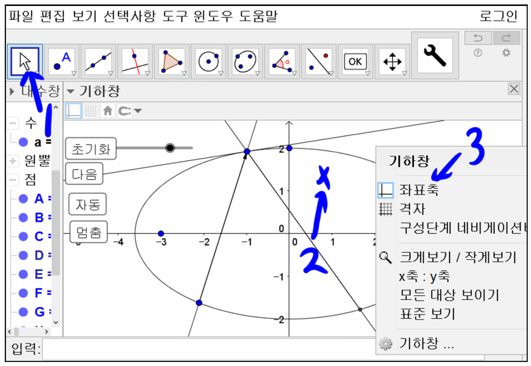
'버튼' 도구를 선택합니다. 마우스를 'x' 표시한 곳을 선택합니다. 캡션에 멈춤

이라고 적고 Geogebra 스크립트에

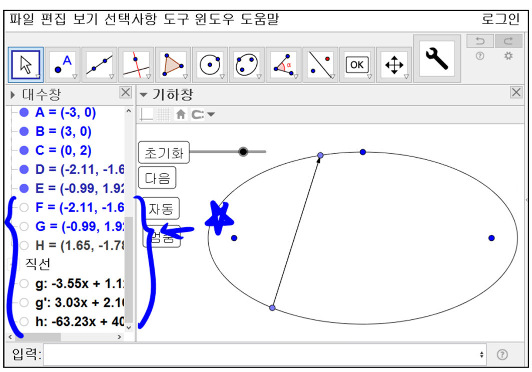
1 번째 줄 : 애니메이션시작[a, false]

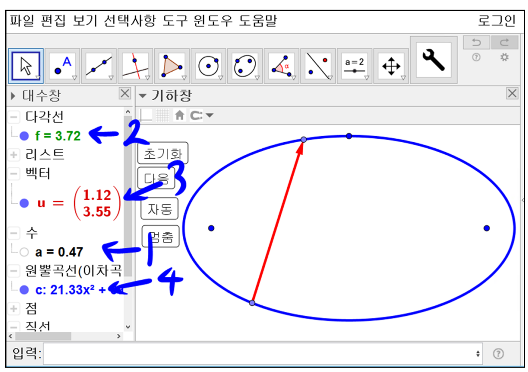
라고 입력합니다. 확인을 선택합니다. 기하창에 초기화, 다음, 자동, 멈춤 버튼을 완성한 상태입니다. 버튼을 눌러 실행하시면 원하는 결과를 얻을 수 있습니다. 이제 다음 그림처럼 최종적으로 꾸며 보겠습니다.



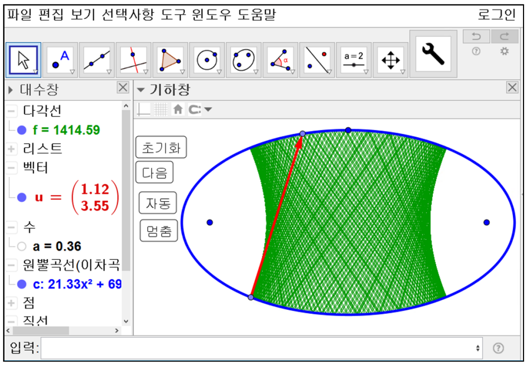


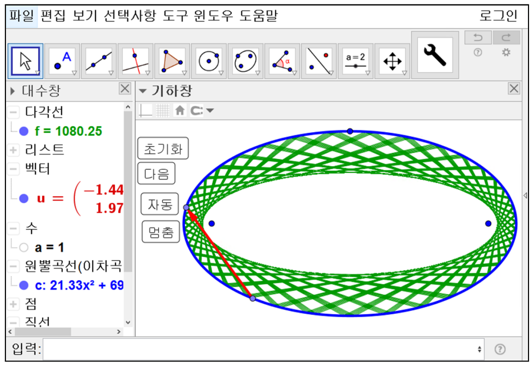
‘이동’ 도구를 선택합니다. 기하창 빈 곳을 선택합니다. [Ctrl]를 누른 상태에서 [A]를 누릅니다. 기하창의 모든 개체가 선택될 것입니다. 마우스 오른쪽 버튼을 클릭합니다. 이름 보이기를 선택합니다. ‘이동’ 도구를 선택합니다. 기하창 빈 곳을 선택합니다. 마우스 오른쪽 버튼을 클릭합니다. 좌표축을 선택합니다.





대수창에 점 F, 점 G, 점 H, 선 g, 선 g′, 선 h 앞에 색칠된 파란색 작은 원을 선택하여 채워지지 않게 합니다. 기하창에서는 동시에 점 F, 점 G, 점 H, 선 g, 선 g′, 선 h 가 보이지 않게 됩니다. 수 a도 마찬가지로 보이지 않게 합니다. 다각선 f의 색을 녹색으로 합니다. 벡터 u의 색을 빨간색으로 합니다. 타원 c를 파란색으로 합니다. 색은 자신의 취향에 따라 바꾸셔도 됩니다.





버튼을 눌러 실행시켜 봅시다. 초점들 사이에 빛이 통과할 때와 초점들 밖으로 빛을 쏘았을 때의 두 가지 경우가 잘 보입니다. 수학에서의 실험을 했습니다.

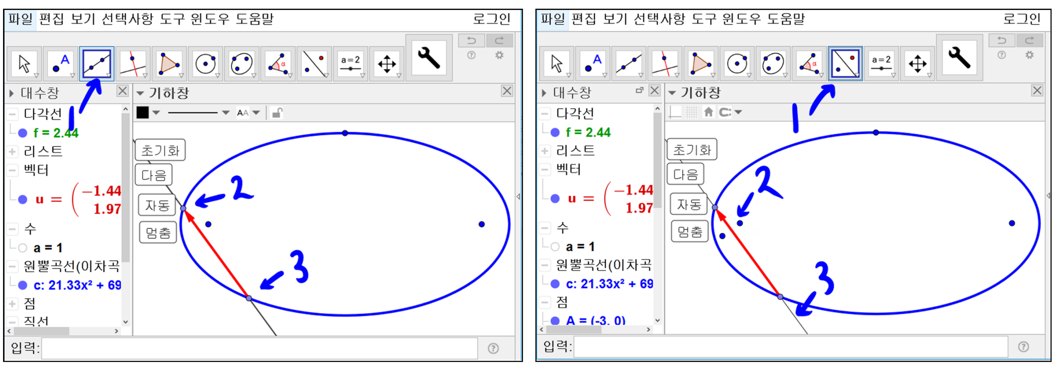
Geogebra 스크립트를 활용한 프로그래밍으로 위의 결과를 만들었습니다. 아마도 프로그램을 처음 접하는 경우에는 조금 힘들 수 있겠습니다. 위의 내용을 처음부터 차근차근 다시 따라가 보고 만들어 보고 각각에 대하여 분석하려고 노력하다 보면 보다 많은 것을 얻으실 수 있습니다. Geogebra의 기본 기능을 활용하여 만드는 데에 있어 그 과정을 보면 논리적 구조가 있습니다. 또한 창의적인 수학적 사고를 통하여 명령어나 기본 도구를 조합 하면 이러한 논리구조를 만들 수 있습니다. 이제 여러분이 가지고 있는 수학적 창의력을 하나씩 수학문제 풀듯이 풀어서 수학적 사고과정을 Geogebra의 기본 기능과 조합하여 무엇인가를 만들어 내면 창의적인 연구에서 원하는 결과물을 얻을 수 있을 것입니다.

1.1.4 타원 내부에서의 빛 반사에 의한 포락선에 대한 예측

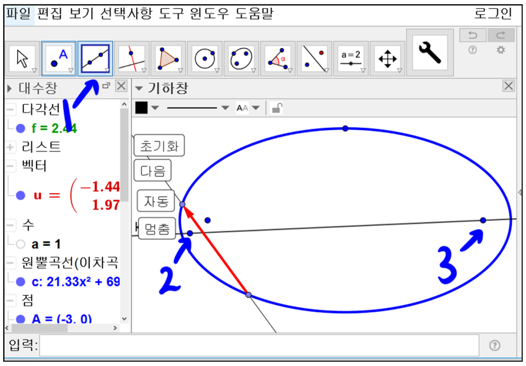
포락선의 정체를 타원 또는 쌍곡선이라고 예측만 한 상태로 2017학년도 경남과학고등학교 수학과 현장연구 특강 강의록이 완성되었고 강의록을 완성하면서 타원에서는 빛 반사에 대한 문제를 다시 정리하는 기회가 되었고 더 깊게 이해가 되었습니다. 문제를 아주 단순하게 바꿔 생각해 보았습니다. 원에서 빛을 반사시킨다면 이 빛이 지나는 곳과 지나지 않는 경계는 즉 포락선이 동심원이 된다는 사실을 생각하고 이러한 성질이 타원에서도 유사하지 않을까 하는 생각에서 다음과 같은 예측을 했습니다.

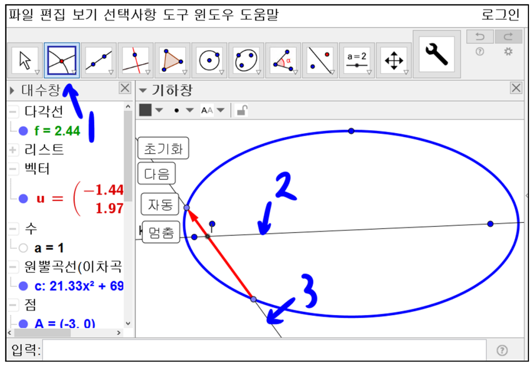
타원 내부에서의 빛 반사에 의한 포락선은 초점이 그대로 유지되는 타원 또는 쌍곡선이 된다.

수학적으로 증명하기 전에, 이 예측이 맞는지에 대한 실험을 지금부터 시도해보겠습니다.

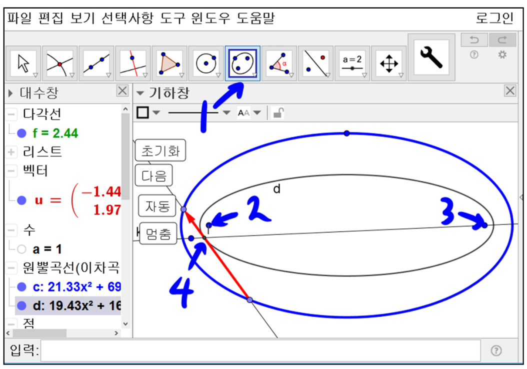


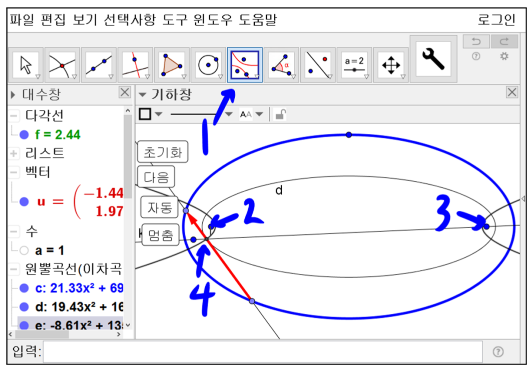
이미 만들어 놓은 자료를 실행시킵니다. '직선' 도구를 선택합니다. 빛의 시점과 종점을 차례로 선택하여 직선을 만듭니다. '직선에 대하여 대칭' 도구를 선택합니다. 초점을 선택하고 직선을 선택합니다. 초점의 직선에 대칭인 점이 그려졌습니다.



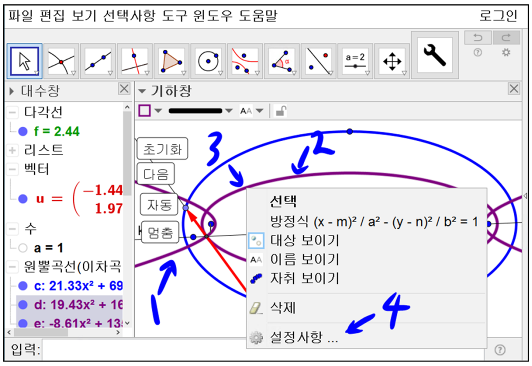


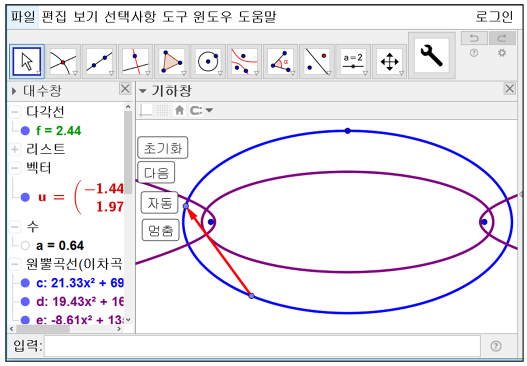
‘직선’ 도구를 선택합니다. 직선에 의해 반사된 점을 선택하고 다른 초점을 선택합니다. 다른 직선이 그려졌습니다. ‘교점’ 도구를 선택합니다. 이 두 직선을 선택합니다. 교점이 그려졌습니다.



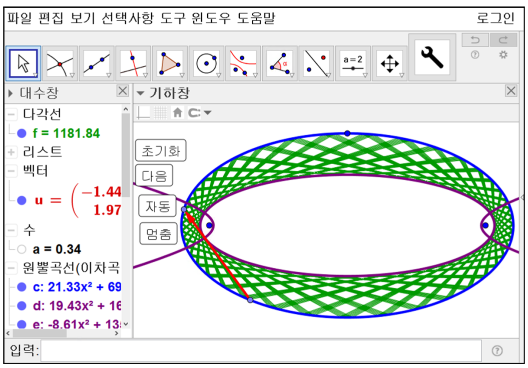


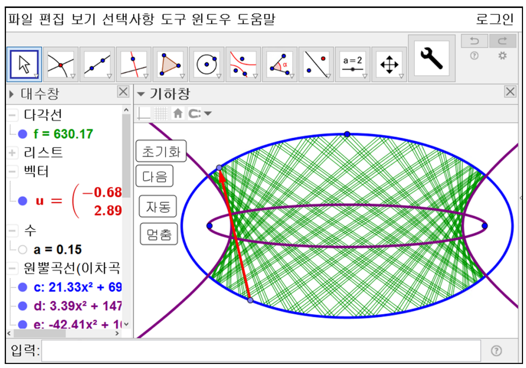
‘타원’ 도구를 선택합니다. 두 초점을 선택하고 교점을 선택합니다. 교점을 지나는 타원이 그려졌습니다. ‘쌍곡선’ 도구를 선택합니다. 두 초점을 선택하고 교점을 선택합니다. 교점을 지나는 쌍곡선 그려졌습니다. 광선을 쏘는 위치에 따라서 교점이 생기지 않는 특수한 경우도 생길 수 있습니다.





[Ctrl]를 누른 상태에서 작은 타원과 쌍곡선을 선택하면 타원과 쌍곡선이 동시에 진해집니다. 계속 [Ctrl]를 누른 상태에서 진한 타원위에 마우스를 위치하고 마우스 오른쪽 버튼을 클릭하면 메뉴가 보입니다. 설정사항에 들어가서 색상을 자주색을 선택하고 스타일에 들어가서 선 굵기를 9로 해줍니다. 타원과 쌍곡선이 굵은 자주색 선이 되었습니다. 이 타원과 쌍곡선을 그리기 위해 만들었던 두 직선과 직선에 대칭한 점, 교점을 보이지 않게 합니다.

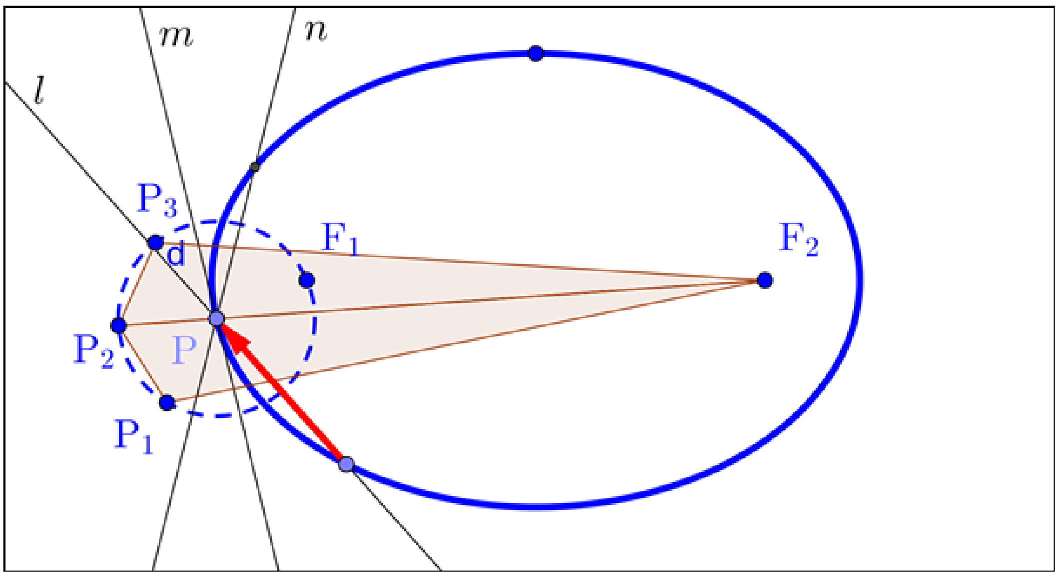




빛을 두 초점 사이 밖에 위치시킨 다음 실행해 봅니다. 작은 타원이 포락선과 딱 맞아 떨어졌습니다. 빛을 두 초점 사이에 위치시킨 다음 실행해 봅니다. 쌍곡선과 포락선이 딱 맞아 떨어졌습니다.

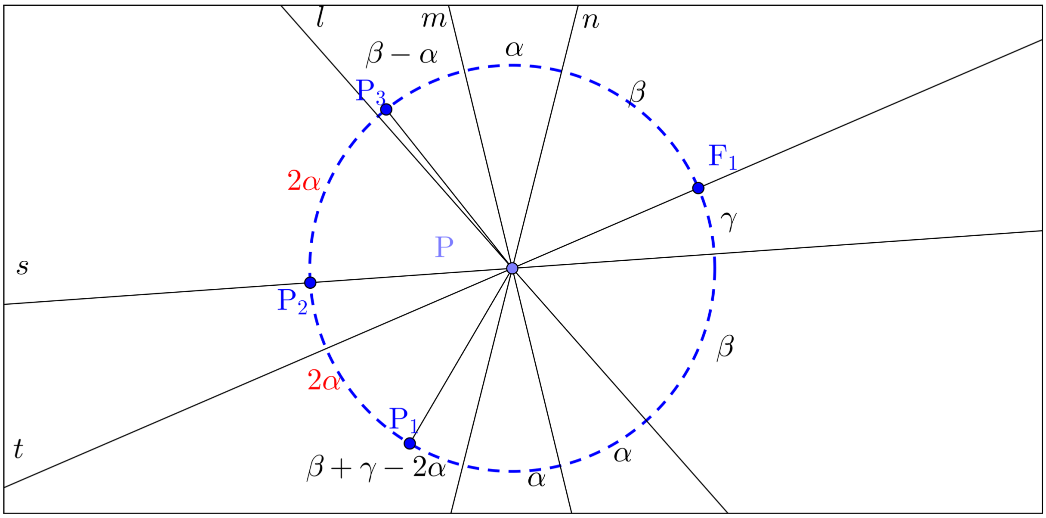
1.1.5 타원 내부에서의 빛 반사에 의한 포락선의 수학적 증명

정말 예측한대로 되었습니다. 아직까지 수학적인 증명은 하지 않고 Geogebra로 결과를 예측했고 예측에 대한 수학적 실험이 성공한 상태입니다. 포락선 자료를 가만히 들여다보며 또 한 주가 지나갔습니다. 처음에는 방정식으로 증명을 해볼까 생각했으나 성공할 수 있겠지만 너무 많은 변수로 일단 다른 방식이 있을까 고민했습니다. 지금까지의 결과가 다 기하적인 그림에 대한 작도로 만든 그림이므로 작도로써 증명이 가능할 것이라고 생각했습니다. Geogebra의 동적 기하를 이용하여 여러 가지 그림을 그려보고 점도 움직여 보며 고민을 했습니다. 여러 가지 고민한 끝에 아래와 같은 그림을 생각하게 되었습니다.

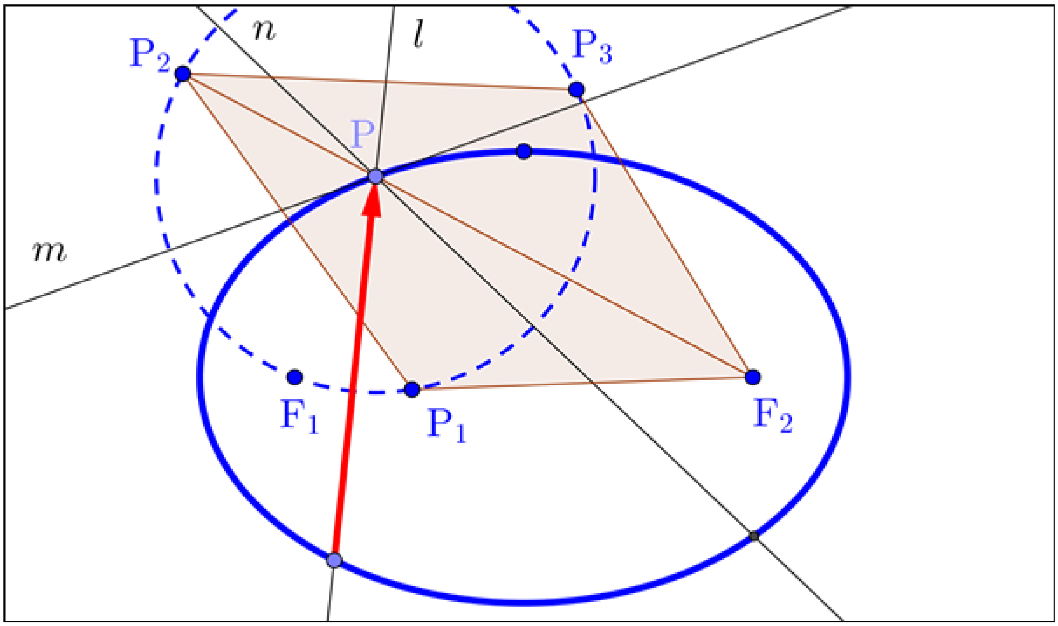


타원의 초점사이 밖으로 빛을 쏜 경우는 위와 같은 그림으로 이유가 설명됩니다. 점 F_1의 점 P에 접하는 타원의 접선 m, 점 P를 향하여 처음 빛을 쏜 직선 l, 그리고 점 P에서 반사된 직선 n에 대한 대칭점을 각각 점 P_2, 점 P_1, 점 P_3라고 하면 타원상의 점 P를 지나는 직선에 다 반사된 점들이므로 이들 반사된 점들은 접점 P에서 같은 거리에 떨어져 있습니다. 이 그림에는 선을 그려 놓지 않았지만 직선 PF_1 과 직선 l이 이루는 각은 당연히 직선 PP_1과 직선 l이 이

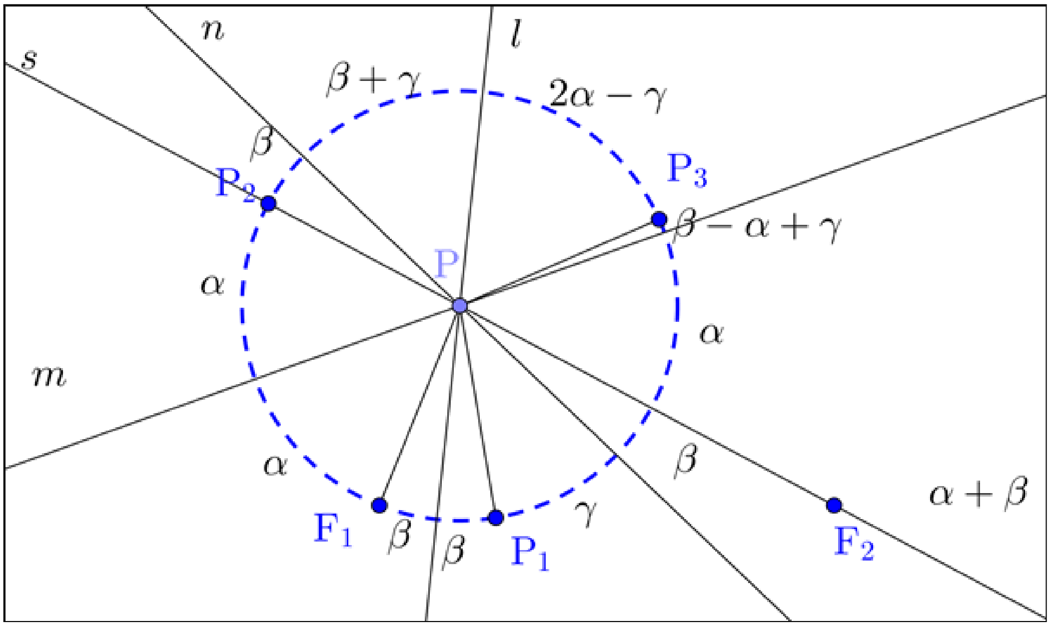
루는 각하고 같습니다. 마찬가지로 직선 $\mathrm{PF_1}$과 직선 m이 이루는 각은 당연히 직선 $\mathrm{PP_2}$과 직선 m이 이루는 각과 같고 직선 $\mathrm{PF_1}$과 직선 n이 이루는 각은 당연히 직선 $\mathrm{PP_3}$과 직선 n이 이루는 각이 같습니다. 점 $\mathrm{F_2}$가 타원의 초점이므로 직선 $\mathrm{PF_1}$과 직선 m이 이루는 각은 당연히 직선 $\mathrm{PF_2}$과 직선 m이 이루는 각하고 같습니다.



점 P의 주위를 확대해서 보이지 않지만 직선 $\mathrm{PP_2}$는 점 $\mathrm{F_2}$를 지나고 있습니다. 이를 잘 활용하여 증명을 하면 그림처럼 각 $\mathrm{P_2PP_3}$와 각 $\mathrm{P_2PP_1}$이 같아지므로 빛이 반사된 후에도 초점 $\mathrm{F_1}$과 초점 $\mathrm{F_2}$에서 직선 l과 반사되어 만들어진 직선 n위의 한 점에서의 최소 거리의 합이 같아집니다. 다시 말해 선분 $\mathrm{P_1F_2}$와 선분 $\mathrm{P_3F_2}$의 길이가 같아집니다. 이 말은 같은 타원에 모든 반사된 선들이 접한다는 말이 되므로 작은 타원이 포락선이 되는 것입니다.



타원의 초점사이로 빛을 쏜 경우는 위와 같은 그림으로 이유가 설명됩니다. 점 F_1의 점 P에 접하는 타원의 접선 m, 점 P를 향하여 처음 빛을 쏜 직선 l, 그리고 점 P에서 반사된 직선 n에 대한 대칭점을 각각 점 P_2, 점 P_1, 점 P_3라고 하면 타원상의 점 P를 지나는 직선에 다 반사된 점들이므로 이들 반사된 점들은 접점 P에서 같은 거리에 떨어져 있습니다. 이 그림에는 선을 그려 놓지 않았지만 직선 PF_1 과 직선 l이 이루는 각은 당연히 직선 PP_1과 직선 l이 이루는 각하고 같습니다. 마찬가지로 직선 PF_1과 직선 m이 이루는 각은 당연히 직선 PP_2과 직선 m이 이루는 각과 같고 직선 PF_1과 직선 n이 이루는 각은 당연히 직선 PP_3과 직선 n이 이루는 각이 같습니다. 점 F_2가 타원의 초점이므로 직선 PF_1과 직선 m이 이루는 각은 당연히 직선 PF_2과 직선 m이 이루는 각하고 같습니다.



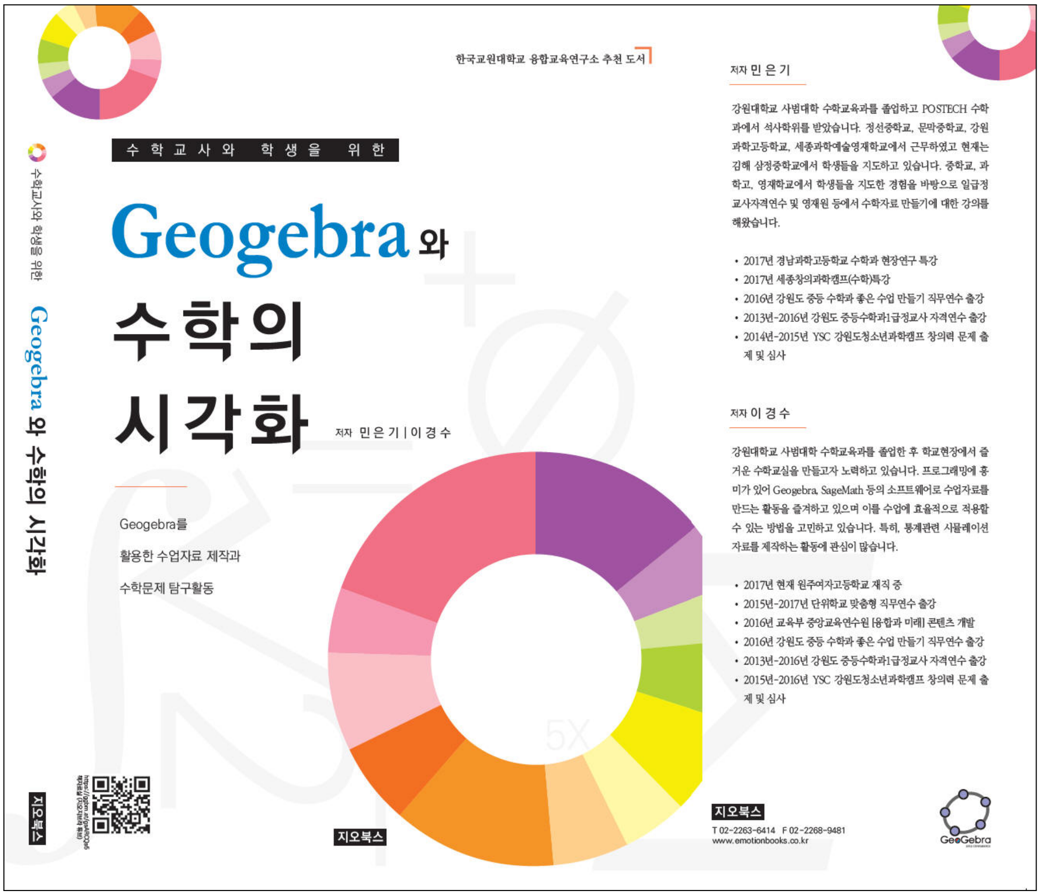
점 P의 주위를 확대해서 보이지 않지만 직선 $\mathrm{PP_2}$는 점 $\mathrm{F_2}$를 지나고 있습니다. 이를 잘 활용하여 증명을 하면 그림처럼 각 $\mathrm{P_2PP_3}$와 각 $\mathrm{P_2PP_1}$이 같아지므로 빛이 반사된 후에도 초점 $\mathrm{F_1}$과 초점 $\mathrm{F_2}$에서 직선 l과 반사되어 만들어진 직선 n위의 한 점에서의 거리의 차의 최대가 같아집니다. 다시 말해 선분 $\mathrm{P_1F_2}$와 선분 $\mathrm{P_3F_2}$의 길이가 같아집니다. 이 말은 같은 쌍곡선에 모든 반사된 선들이 접한다는 말이 되므로 작은 쌍곡선이 포락선이 되는 것입니다.

이 결과물을 덕문여자고등학교 정봉균 선생님이 보시고 흡족해 하셨고 저도 또한 이러한 연구과정의 결과물을 만들 기회를 가지게 되어 만족스러웠습니다. 여러분들도 정봉균 선생님과 민은기 선생님과의 이야기를 통하여 많은 영감을 가졌으면 하는 바램입니다. 앞으로 여러분도 수학에서의 기하에 대한 연구를 한다면 Geogebra를 활용해 보시기 바랍니다.

찾아보기

그동안 했던 강의 자료 중 일부를 책으로 엮음.

http://min7014.iptime.org/math/2017063002.htm



https://ggbm.at/gsARCQs5

책자료실(지오지브라 튜브)

[참고]

[민은기 선생님의 수학자료실]

Homepage : http://min7014.iptime.org

Facebook Page : https://www.facebook.com/mineungimath

YouTube Channel : https://goo.gl/JpzU5i

[이경수 선생님 블로그]

http://blog.naver.com/evening07

[GeoGebra 5.0.363.0-3D (03 June 2017) 설치파일]

Installer : https://goo.gl/YvjsCV (From Home Page)

Installer : https://goo.gl/n69yEl (From Google Drive)

[GeoGebra 5.0.462.0-d (02 May 2018) 설치파일]

Installer : https://goo.gl/SsdFBd (From Home Page)

Portable : https://goo.gl/FxJxES(From Home Page)

Installer : https://goo.gl/dqtbfk (From Google Drive)

Portable : https://goo.gl/zwundc(From Google Drive)